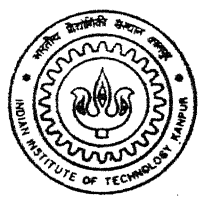
Segmentation and Performance Evaluation of Steel Defect Images

by
Swaroop K Chalasani



DEPARTMENT OF MECHANICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR
March, 2000

Segmentation and Performance Evaluation of Steel Defect Images

A Thesis Submitted
in Partial Fulfillment of the Requirements
for the Degree of
Master of Technology

by

Swaroop K Chalasani

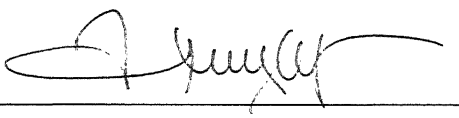
to the

Department of Mechanical Engineering
Indian Institute of Technology, Kanpur

March, 2000

Certificate

This is to certify that the work contained in the thesis entitled "*Segmentation and Performance Evaluation of Steel Defect Images*", by *Swaroop K Chalasani*, has been carried out under our supervision and that this work has not been submitted elsewhere for a degree.



March, 2000

(Dr. Amitabha Mukerjee)
Department of Computer Sc. and Engg.,
Indian Institute of Technology,
Kanpur.

Acknowledgment

I express my deep sense of gratitude to my thesis advisor Dr. Amitabha Mukerjee for his excellent guidance, invaluable suggestions, and encouragement during the tenure of this work. He is even not short of providing fun for his students. Much of his informality put a shy person like me at great ease. I am also thankful to RDCIS, Steel Authority of India Ltd. for supporting this work.

Dr. Bhaskar Dasgupta has been timely helpful in MATLAB and LaTeX and I am really thankful for him. An Ideal atmosphere has been created to work in Robotics Lab by various people and facilities and I am thankful to the Head of Robotics Lab for kindly allowing me to work in such a wonderful lab. I wish the lab prosper more in future.

I wish to acknowledge all the friends, who had provided a wonderful company, which makes cherish my stay at IIT Kanpur. I thank every of the person who is directly or indirectly involved in helping me out in any way.

Abstract

Automatic metal surface inspection in automation is being investigated for two decades, but there is no general approach available for recognition of defects automatically. Different tasks involved in this challenge are, image acquisition, segmentation of anomalies, anomaly measurement, feature extraction and decision making for classification of defects. Our work focuses on segmentation, i.e. isolating the suspected anomalies in the image. In the recent years watershed algorithms, based on mathematical morphology has proved very effective and computationally very efficient for this task, but its effectiveness in steel inspection has not been explored. We report a series of investigations on several defect types using this approach and demonstrate the effectiveness in isolating the defects (segmentation). Another contribution of this work is the use of *groundtruth* or validated correct results in evaluating the algorithms. Selection of segmentation parameters as reported is done manually, but with the performance evaluation approach, this work can be extended towards unsupervised (autonomous segmentation).

Contents

List of Figures

Chapter 1

Introduction

Automatic metal surface inspection is a well known problem and is being investigated for more than two decades. In Steel rolling mills the surface quality of strip is currently controlled mainly by human on-line visual inspection before cutting the strip into variable length sheets for delivery. Human inspectors classify the defects according to their cause and origin because the inspection results are used as feedback to correct the manufacturing process. The experience of the inspector is essential, because there are seldom any fixed defect criteria. The inspector's pass/reject decisions seem to be based on the types of defects and their extent, the maximum number of defects per unit of surface area and the total number of defects on the entire inspected strip. In addition, the inspector's knowledge of the customer and the use of the strip have a great impact on the decisions. The main purpose of the automatic surface inspection is to *grade* it in quality, enabling manufacturer to deliver right quality of steel to the customer. At the same time, he can provide the information of defect location at various places, apart from improving the process itself. Metal strip is typically a 0.3 - 2.0m wide and 0.1-5.0mm thick shiny metallic band and endless length continuously rolling in-between rollers going through various sections undergoing different manufacturing transformations. It is basic material in the fabrication of a large variety of products e.g. in electronic and electrical, automotive and construction industries. This naturally subjects the strip to automatic forming and surface finishing steps that require correct metallurgical and surface properties.

Automation of the visual inspection of metal strip is an acute problem because the human visual inspection is an unreliable, tedious and boring task. Several automatic

inspection systems have been designed for this purpose, but they are mainly defect detection devices. Defect recognition and assessment are performed by a human operator that is notified of discrepancies found. The computational requirements of strip inspection are severe. In a typical strip case, the strip to be inspected is 1m wide and moves at the speed of 1.5m/s and the required minimum size of the defects to be detected is of the order of one millimeter. This amounts to over one and half million pixels per second depending on the resolution of the CCD camera [23].

One of the earliest work on metal surface inspection was [9] in 1979, where a feasibility study was presented for automatic surface inspection system for Flat Rolled Steel. The camera setup assumed was television camera. In [8] the feasibility of applying image processing techniques to metal surface inspection is demonstrated. The method chosen was based on pattern recognition techniques to classify metal surfaces into classes of different roughness. A two-level tree classifier using non-parametric linear classifier at each node was used for classification of defects. One of the work employing modern facilities was [23], in which a proto-type of an automated visual on-line metal strip inspection system, employing CCD camera and capable of both detecting and classifying the surface defects of copper alloy strip was outlined. Detecting a defect means to say whether a defect is there or not and recognition is to say what kind of defect it is. The inspection algorithms used in this model were based on morphological pre-processing and combined statistical and structural defect recognition. A real-time setup was described in [3], where a hardware setup for 560 Kpixels/sec using off-the-shelf components was described. Segmentation used was adaptive thresholding followed by a tree classification. A 4-defect identification system was implemented and reported by Nema [21]. This problem was also studied from Content Based Image Retrieval (CBIR) point of view by Ashok [16]. One of the recent approaches employing morphological segmentation for industrial inspection was [18] for measurement of droplet size distribution (granulometry). Our work is much more complex than this one as it involves many kinds of defects digitized under various lighting conditions. The first step in the approach for solving such a problem is to have a good atlas of all the possible defects, the system is expected to recognize. We considered the atlas [1] prepared by Nema [21] as a base for design of our system.

This work focuses on a part of the visual inspection system. The specific tasks involved are image acquisition, segmentation, feature extraction, and classification. Among these we investigated the segmentation, which is very critical in the whole system. We

[1] http://www.iitk.ac.in/robotics/project_lists/sail/atlas.html

discuss about this in greater detail in chapter 2.

1.1 Overview of the Experimental System

The inspection system prototype employs an image acquisition system that is sensitive to most strip discrepancies. Image acquisition is followed by a preprocessing stage based on grey-scale morphological erosion and dilation operations, noise removal filters that enhance the interesting features and suppress noise. The preprocessed image data is subjected to various morphological transformations which we describe in later chapters for analysis and classification.

All the defect candidates are submitted to an automatic defect analysis stage. The recognition is performed with a tree classifier by using the size, shape, and orientation features computed from the segmentation and other stages.

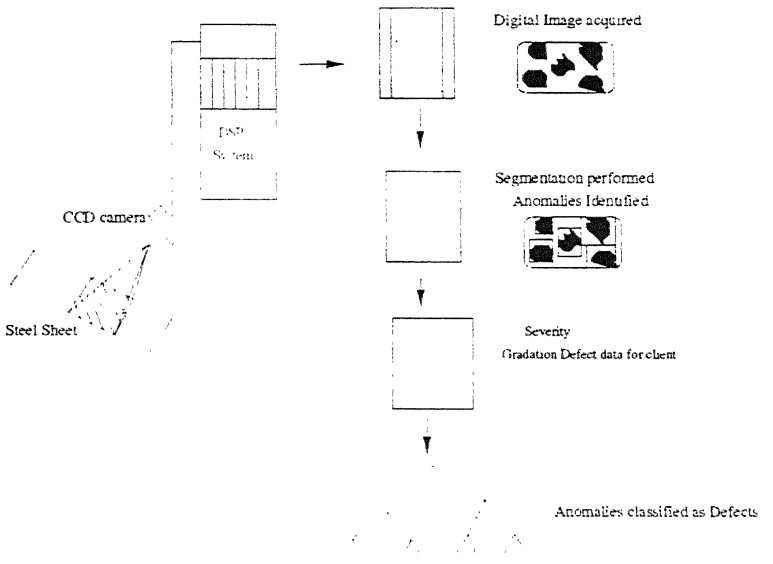


Figure 1.1: Surface Inspection Configuration: This work focuses on segmentation where each defect feature is isolated in the image

The configuration of the Surface Inspection is shown in 1.1. Various stages of the experimental system are as follows [4]:

- Image Acquisition

- Segmentation (Anomalies)
- Feature Extraction
- Classification (Defects)

These are discussed in detail in the following sections.

1.2 Image Acquisition

A good Image Acquisition system is vital for setting up a Machine Vision system. This is however an iterative process. A first set of images were captured to acquaint ourselves with the types of anomalies occurring and the imaging conditions in which the system is expected to operate. These were captured on-site at Bokaro Steel Plant largely in Ambient lighting. These images were analyzed and observing the illumination requirements for a few highly occurring defects such as Iron Particles, Black Patches, Roll in Scales, Scratches,holes etc. a new illumination system was set-up. A second set of images was then taken as a means to test the illumination system, in laboratory.

1.2.1 Set-up of Image Acquisition System

Successful Image Understanding depends on the quality of information present in the images. This in turn, is highly dependent on the quality of the lighting, optics and sensors used to capture the image.

In this section we present the design procedure for the Image Acquisition system, for satisfactory Identification of defects. The system requirements are stated on the basis of the object surface characteristics such as the luster of the surface, whether stationary or moving, are a of the surface to be imaged etc. The test images were captured with the designed set-up. The notion of these images being "good" or "better" however was still based on subjective means. There is a large scope of improvements in our set-up. The image acquisition system design is an iterative process - the bottlenecks are not discovered until well into the testing stage. The idea was to understand the parameters of the image acquisition system and obtain a basis for setting up the same for the On-line system operating in real-time. This was discussed in great detail in chapter 2 of Nema [21].

1.3 Segmentation

The first step in *image analysis* generally is to segment the image. Segmentation subdivides an image into its constituent parts or objects. The level to which this subdivision is carried depends on the problem being solved. That is, segmentation should stop when the objects of interest in an application have been *isolated.* In defect isolation in, surface inspection the first step is to segment the background from the defect and then to segment the defects from one another. There is no point in carrying segmentation below this scale.

In general, autonomous segmentation (unsupervised segmentation) is one of the most difficult tasks in image processing. This step is the process determines the eventual success or failure of the analysis. In fact, effective segmentation rarely fails to lead to a successful solution. For this reason, considerable care should be taken to improve the probability of rugged segmentation.

Segmentation algorithms for monochrome images generally are based on one of two basic properties of gray-level values: discontinuity and similarity. In the first category, the approach is to partition an image based on abrupt changes in gray-level. The principal areas of interest within this category are detection of isolated points and detection of lines and edges in an image. The principal approaches in second category are based on thresholding, region growing, and region splitting and merging. The concept of segmenting an image based on discontinuity and similarity of the gray-level values of its pixels is applicable to both static and dynamic (time varying) images. In the latter case, however, motion can often be used as powerful cue to improve the performance of segmentation algorithms. We have investigated morphological based segmentation through watersheds [28] with a number of modifications like applying preprocessing based on Morphology, Multiscale gradient etc. These are discussed in section 1.7 and Chapter 2. The reason for choosing morphological approach are numerous. The prime reason is they deal directly with shape, which makes it easy for various tasks like segmentation of a particular shape, measurement of a geometric feature, shape manipulation etc. The second reason is they are computationally faster than any other transformations required for a given task. In particular for segmentation task, a summary of comparisons among various methods is reported in [28]. Also this can be substantiated by the fact that a number of leading machine vision vendors (see footnote in section 1.7) have implemented morphological techniques in their products.

1.4 Feature Extraction

After segmentation, the next step is to extract some suitable feature, which can distinguish various kinds of defects in some feature space. The feature extraction although predominantly done after segmentation, but its not limited to do so. Some of the features based on histogram, should be extracted before the segmentation. Also features based on color should be extracted before the conversion of image to grey-level. Typical features in steel surface inspection application are

- Geometric Measures:
  - length
  - width
  - area
  - circularity
  - aspect ratio
- Occurrence:
  - certain grades
  - specific times of year
  - during certain processing conditions
- Luminance Measures:
  - light
  - dark
  - contrast
- Anomaly Distribution:
  - Clustering of objects together
  - location within the field of view

These are extracted at various stages of image undergoing transformations. This feature extraction is used for the following reasons

- Reduction of data: e.g. thousands of pixels in an image to a list of a few measurements.
- Should be less sensitive or invariant to noise.
- Should contain fewer redundancies.

1.4.1 Selection of Features

The selection of features is again a subjective matter depending on the application.

The feature extraction demands a good mathematical background to reduce the dimensionality of the original feature data set from which a set of new composite variables are determined. Also transformations should be devised which maximize variations between categories. Generally Mean and Co-variance matrices are employed. A good example is Principle Components Analysis (PCA). A defect image is then characterized by a set of features in a feature vector. The feature space should be compact, relatively invariant for each defect type and permit easy distinction among different defects (Figure 1.2).

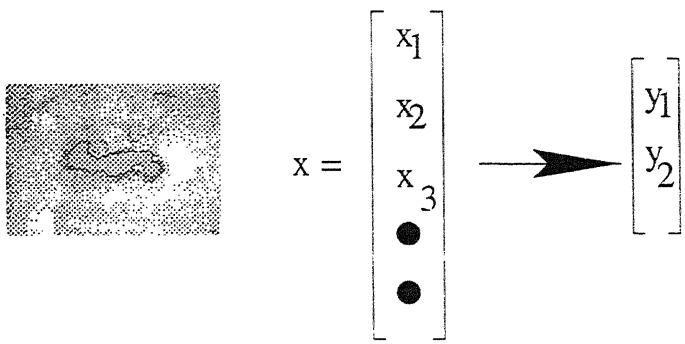


Figure 1.2: Feature Mapping. Millions of pixels → a vector of two measurements

1.5 Classification

As stated in above section, the ability to perform the pattern recognition at some level is fundamental to image analysis. Here a pattern or feature vector is a quantitative or structural description of an object or some other entity of interest in an image. In general, a pattern is formed by one ore more descriptors, such as those discussed in section 1.4.

The input to the classifier is a feature vector and the task of the classifier is to compare this feature vector with some set of known or pre-defined reference class vectors to assess similarity. The defect is then defined as the class with the closest association to the feature vector. The task would be easy if we already knew the reference class vectors and if each was well defined. Then we could use simple measures of association to determine class membership. Some measures of similarity are [?]:

- Association
- Correlation
- Distance
- Probabilistic
- Functional

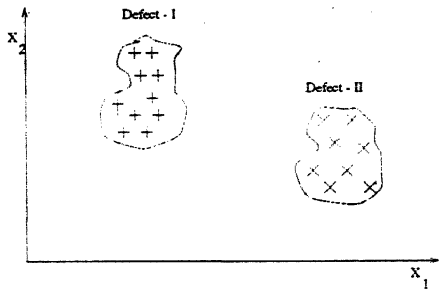


Figure 1.3: Classification in Feature space

1.6 Classification

As stated in above section, the ability to perform the pattern recognition at some level is fundamental to image analysis. Here a pattern or feature vector is a quantitative or structural description of an object or some other entity of interest in an image. In general, a pattern is formed by one ore more descriptors, such as those discussed in section 1.4. The input to the classifier is a feature vector and the task of the classifier is to compare this feature vector with some set of known or pre-defined reference

Also cluster analysis can be applied to find natural groupings or structures in data. Some well known classifiers include minimum distance classifier, matching by correlation, statistical classifier, Bayes classifier, Neural networks etc.

1.7 Mathematical Morphology

Mathematical morphology provides an approach to the processing of digital images which is based on shape. Appropriately used, mathematical morphological operations tend to simplify image data preserving their essential shape characteristics and eliminating irrelevancies. As the identification of defects, defect features correlate directly with shape, mathematical morphology became very popular in image analysis for machine vision recognition [11].

Morphological tools are implemented in most advanced image analysis packages like HALCON[2], MMORPH[3],ProSIGMASCAN[4], etc..which are very often used in applications where shape of objects and speed is an issue like on-line industrial inspection apart from analysis of microscopic images (in biology, material science, geology, and criminology), industrial inspection(gauging, quality assurance), optical character recognition(OCR),and document analysis etc.

The non-morphological approach to image processing are based on not a single method, are based on linear transformations such as convolution, whereas mathematical morphology uses tools of non-linear algebra and operates with point sets, their connectivity and shape. Morphological operations simplify images, and quantify and preserve the main shape characteristics of objects.

Morphological operations are used predominately for the following purposes:

- Image pre-processing (noise filtering, shape simplification with alternate sequential filters)
- Enhancing object structure (skeletonizing, thinning, thickening, convex hull, object marking)
- Segmenting objects from the background (top-hat, watershed, reconstruction)

[2]http://www.mvtec.com
[3]http://www.mmorph.com
[4]http://www.spss.com

- Quantitative description of objects (area, perimeter, projections)

Mathematical morphology exploits point set properties, results of integral geometry, and topology. The initial assumption states that real images can be modeled using **point sets** of any dimension(e.g., N-dimensional Euclidean space); Computer vision uses the digital counterpart of Euclidean space–sets of integer pairs ($\in \mathcal{Z}^2$) for binary image morphology or sets of integer triples ($\in \mathcal{Z}^3$) for gray-scale morphology or binary 3D morphology. Computer vision uses the digital counterpart of Euclidean space–sets of integer pairs ($\in \mathcal{Z}^2$) for binary image morphology or sets of integer triples ($\in \mathcal{Z}^3$) for gray-scale morphology or binary 3D morphology.

A **morphological transformation** Ψ is given by the relation of the image (point set X) with another small point set B called a **structuring element.** B is expressed with respect to a local origin $\mathcal{O}$ (called the representative point). Some typical structuring elements are box, cross etc. are shown in Fig 1.4

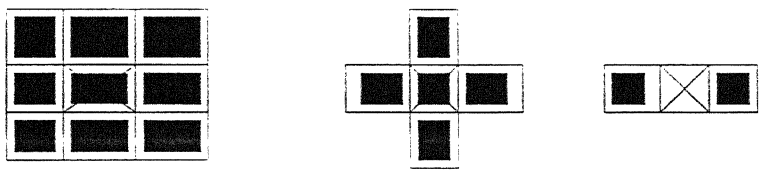


Figure 1.4: Typical Structuring Elements

To apply the morphological transformation $\Psi(X)$ to the image X means that the structuring element B is moved systematically across the entire image. Assume that B is positioned at some point in the image; the pixel in the image corresponding to the representative point $\mathcal{O}$ of the structuring element is called the *current* pixel. The result of the relation between the image X and the structuring element B in the current position is stored in the output image in the current image pixel position.

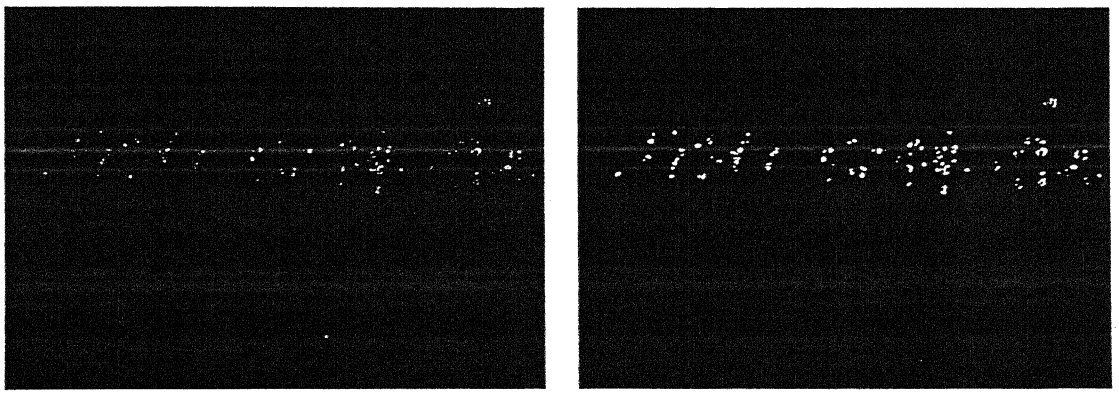
1.8 Binary Dilation and Erosion

The sets of black and white pixels constitute a description of a binary image. Assume that only black pixels are considered, and the white pixels are treated as a background. The primary morphological operations are dilation and erosion, and from these two, more complex morphological operations such as opening, closing and shape decomposition can be constituted. We present them here using Minkowski's formalism [11].

1.8.1 Dilation

The morphological transformation **dilation** $\oplus$ combines two sets using vector addition (or Minkowski set addition, e.g., $((a,b)+(c,d)=(a+c,b+d))$. The dilation $X \oplus B$ is the point set of all possible vector additions of pairs of elements, one from each of the sets X and B.

$$X \oplus B = \left\{p \in E^2 | p = x + b, x \in X \wedge b \in B\right\} \tag{1}$$



(a) (b)

Figure 1.5: Dilation of pinch mark image (a) Binarized Pinch mark (b) Dilation with structuring element of 3 × 3 box type

Dilation with an isotropic 3 × 3 structuring element might be described as a transformation which changes all background pixels neighboring the object pixels. Dilation is an *increasing* transformation. Dilation is used to fill small holes and narrow gulfs in objects. It increases the object size–if the original size needs to be preserved, then dilation is combined with erosion.

1.8.2 Erosion

Erosion $\ominus$ combines two sets using vector subtraction of set elements and is the dual operator of dilation. Neither erosion nor dilation is an invertible transformation.

$$X \ominus B = \left\{p \in E^2 | p + b \in X \quad \forall b \in B\right\} \tag{2}$$

This formula says that every point p from the image is tested; the result of the erosion is given by those points p for which all possible $p + b$ are in X.

(a) (b)

Figure 1.6: Erosion of pinch mark image (a) Binarized Pinch mark (b)Erosion

Fig 1.6 shows the erosion 3×3 element of the same original as in Fig 1.5. Notice that single-pixel-wide lines disappear. Erosion (such as Figure 1.6) with an isotropic structuring element is called *shrink* or *reduce* by some authors. Basic morphological transformations can be used to find the contours of objects in an image very quickly. This can be achieved, for instance, by subtraction from the original picture of its eroded version. Erosion is used to simplify the structure of an object–objects or their parts with width equal to one will disappear. It might thus decompose complicated objects into several simpler ones. Interesting property of erosion(in contrast to dilation) is not commutative.

1.8.3 Hit-or-miss transformation

The hit-or-miss transformation is the morphological operator for finding local patterns of pixels, where local means the size of the structuring element. It is a variant of template matching that finds collections of pixels with certain shape properties (such as corners, or border points). This transformation is especially important in thinning and thickening of objects. Operations described hitherto used a structuring element B, and we have tested points for their membership of X; we can also test whether some points do not belong to X. An operation may be denoted by a pair of disjoint sets $B = (B_1, B_2)$, called a **composite structuring element**. The **hit-or-miss** transformation $\otimes$ is defined as

$$X \otimes B = \left\{x | B_1 \subset X \wedge B_2 \subset \overline{X}\right\} \tag{3}$$

This means that for a point x to be in the resulting set, two conditions must be fulfilled

simultaneously. First the part B_1 of the composite structuring element that has its representative point at x must be contained in X, and second, the part B_2 of the composite structuring element must be contained in $\overline{X}$. The hit-or-miss transformation operates as a binary matching between an image X and the structuring element(B_1, B_2). It may be expressed using erosion and dilations as well

$$X \otimes B = (X \ominus B_1) \cap (\overline{X} \ominus B_2) = (X \ominus B_1)\,(X \ominus B_2)$$

1.8.4 Opening and Closing

Erosion and dilation are not inverse transformations–if an image is eroded and then dilated, the original image is not re-obtained. Instead, the result is a simplified and less detailed version of the original image. Erosion followed by dilation creates an important morphological transformation called **opening**. The opening of an image X by the structuring element B is denoted by $X \circ B$ and is defined as

$$X \circ B = (X \ominus B) \oplus B$$

Dilation followed by erosion is called closing. The closing of an image X by the structuring element B is denoted by $X \bullet B$ and is defined as

$$X \bullet B = (X \oplus B) \ominus B$$

If an image X is unchanged by opening with the structuring element B is called *open with respect to*B. Similarly, if an image X is unchanged by closing with B, it is called *closed with respect to*B. Opening and closing with an isotropic structuring element is used to eliminate specific image details smaller than the structuring element–the global shape of the objects is not distorted. Closing connects objects that are close to each other, fills up small holes, and smoothes the object outline by filling up narrow gulfs. A significant fact is that the iteratively used openings and closings are **idempotent**, meaning that reapplication of these transformations does not change the previous result. Formally,

$$\begin{aligned} X \circ B &= (X \circ B) \circ B \\ X \bullet B &= (X \bullet B) \bullet B \end{aligned}$$

1.9 Gray-scale Dilation and Erosion

Binary morphological operations acting on binary images are easily extendible to gray-scale images using the 'min' and 'max' operations. Erosion (respectively, dilation) of an image is the operation of assigning to each pixel the minimum (maximum) value found over a neighborhood of the corresponding pixel in the input image. The structuring element is more rich than in the binary case, where it gave only the neighborhood. In the gray-scale case, the structuring element is a function of two variables that specifies the desired local gray-level property. The value of the structuring element is added (subtracted) when the maximum (or minimum) is calculated in the neighborhood.

This extension permits a **topographic view** of gray-scale images–the gray-level is interpreted as the height of a particular location of a hypothetical landscape. Light and dark spots in the image correspond to hills and hollows in the landscape. Such a morphological approach permits the location of global properties of the image, i.e., to identify characteristic topographic features on images as valleys, mountain ridges (crests), and watersheds.

1.9.1 Top surface, umbra, and gray-scale dilation and erosion

Consider a point set A in n-dimensional Euclidean space, $A \subset E^n$, and assume that the first $(n-1)$ co-ordinates of the set constitute a spatial domain and the n^{th} co-ordinate corresponds to the value of a function or functions at a point ($n = 3$ for gray-scale images). This interpretation matches the topographic view for a 2D Euclidean space, where points are given by triples of co-ordinates; the first two co-ordinates locate the position in the 2D support set and the third co-ordinate gives the height. The **top surface** of a set A is a function defined on the $(n-1)$-dimensional support. For each (n-1)-tuple, the top surface is the highest value of the last co-ordinate of A. If the space is Euclidean the highest value means supremum. Let $A \subseteq E^n$ and the support $F = x \in E^{n-1}$for some$y \in E, (x, y) \in A$. The **top surface** of A, denoted by $T[A]$, is a mapping $F \rightarrow E$ defined as

$$T[A](x) = \max\{y, (x, y) \in A\}$$

The next concept is the **umbra** of a function f defined on some subset F (support) of $(n-1)$-dimensional space. The usual definition of umbra is a region of complete shadow resulting from obstructing the light by a non-transparent object. In mathematical morphology, the umbra of f is a set that consists of the top surface of f and everything

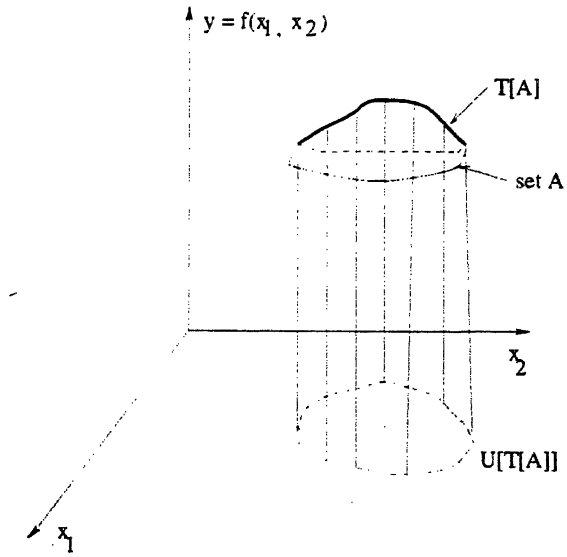


Figure 1.7: Umbra of the top surface of a set is the whole subspace below it

below it. Formally, let $F \subseteq E^{n-1}$ and $f : F \to E$. The **umbra** of f, denoted by $U[f], U[f] \subseteq F \times E$, is defined by

$$U[f] = \{(x, y) \in F \times E, y \leq f(x)\}$$

We will now define the gray-scale dilation of two functions as the top surface of the dilation of their umbras. Let $F, K \subseteq E^{n-1}$ and $f : F \to E$ and $k : K \to E$. The **dilation** $\oplus$ of f by k, $f \oplus k : F \oplus K \to E$ is defined by

$$f \oplus k = T\{U[f] \oplus U[x]\}$$

Notice here that $\oplus$ on the left-hand side is dilation in the gray-scale image domain, and $\oplus$ on the right-hand side is dilation in the binary image. A new symbol was not introduced since no confusion is expected; This definition explains what gray-scale dilation means, but does not give a reasonable algorithm for actual computation, which is as follows

$$(f \oplus k)(x) = \max\{f(x - z) + k(z), z \in K, x - z \in F\} \quad (4)$$

The definition of **gray-scale erosion** is analogous to gray-scale dilation. The gray-scale erosion of two functions (point sets)

1. Takes their umbras

2. Erodes them using binary erosion

3. Gives the result as the top surface

The computational definition of erosion is as follows

$$(f \ominus k)(x) = \min \{f(x+z) - k(z), z \in K, x+z \in F\} \tag{5}$$

with usual notation. We present a morphological pre-processing on a pinch mark image with a lot of noise and uneven background: the aim is to reduce noise and locate individual cells.

Gray-scale opening and closing is defined in the same way as in the binary morphology. There is a simple geometric interpretation of gray-scale opening. the opening of f by structuring element k can be interpreted as sliding k on the landscape f. The position of all highest points reached by some part of k during the slide gives the opening, and a similar interpretation exists for erosion.

Gray-scale opening and closing is often used in applications to extract parts of gray-scale image with given shape and gray-scale structure.

1.9.2 Top hat transformation

The top hat transformation is used as a simple tool for segmenting objects in gray-scale images that differ in brightness from the background, even when the background is of *uneven* gray-scale. the top hat transform is superseded by the watershed segmentation for more complicated backgrounds. Assume a gray-level image X and a structuring element K. The residue of opening as compared to original image $X - (X \circ K)$ constitutes a new useful operation called a **top hat transformation**. The top hat transformation is a good tool for extracting light objects (or, conversely, dark ones, of course) on a dark (or light) but slowly changing background. Those parts of the image that cannot fit into structuring element K are removed by opening. Subtracting the opened image from the original provides an image where removed objects stand out clearly. The actual segmentation can be by simple thresholding. The origin of the name is from the fact that, an image were a hat, the transformation would extract only the top of it, provided that the structuring element is large than the hole in the hat.

1.9.3 Morphological Gradient

The basic morphological gradient, also called *Beucher* gradient, is defined as the arithmetic difference between the dilation and the erosion with the elementary structuring element B of the considered grid.

$$\rho_B = \delta_B - \varepsilon_B \tag{6}$$

1.9.4 Alternating Sequential Filtering

Alternative Sequential Filtering creates the image y by filtering the image f by n iterations of the close and open alternating sequential filter characterized by the structuring element b. The sequence of opening and closing can be 'OC', 'CO', 'OCO'and 'COC'. It simplifies the object by removal of noise, and is used as a preprocessing step for finding gradient.

1.9.5 Geodesic transformations

Geodesic methods modify morphological transformations to operate only on some part of an image. For instance, if an object is to be reconstructed from a binary image called marker, say a nucleus of a cell, it is desirable to avoid growing from a marker outside the cell. Another important advantage of geodesic transformations is that the structuring element can vary at each pixel, according to the image. The basic concept of geodesic methods in morphology is geodesic distance. the path between two points is constrained within some set. Suppose that a traveler seeks the distance between London and Tokyo– the shortest distance passes *through* the Earth, but obviously the geodesic distance that is of interest to the traveler is constrained to the Earth's surface.

The **geodesic distance** $d_X(x, y)$ is the shortest path between two points x, y while this path remains entirely contained in the set X. If there is no path connecting points x, y, we set the geodesic distance $d_X(x, y) = +\infty$. This definition is illustrated in Fig 1.8

The **geodesic influence zone** $iz_A(B_i)$ of a connected component B_i of B in A is the locus of the points of A whose geodesic distance to B_i is smaller than their geodesic distance to any other component B. This concept is illustrated in Fig. 1.9. Those points of A which belong to boundary between two geodesic zone contain the *skeleton by influence zones (SKIZ)* of B inside A.

Figure 1.8: The geodesic distance between x and y inside A is the infimum of the length of the paths between these two points which are totally included in A

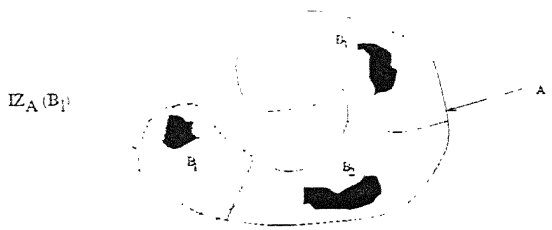


Figure 1.9: Geodesic influence zone of connected component B_1 inside set A

We suppose that the geodesic SKIZ is always made of lines having one pixel thickness and thus separating the different geodesic influence zones.

1.9.6 Morphological reconstruction

Assume that we want to reconstruct objects of a given shape from a binary image that was originally obtained by thresholding. All connected components in the input image constitute the set X. However, only some of the connected components were marked by markers that represent the set Y. This task and its desired result are shown Successive geodesic dilations of the set Y inside the set X enable the reconstruction of the connected components of X that were initially marked by Y. When dilating (δ_x from the marker, it is impossible to intersect a connected component of X which did not initially contain a marker Y; such components disappear. Geodesic dilations terminate when all connected component sets X previously marked by Y are reconstructed, i.e., idempotency is reached:

$$\forall n > n_0, \delta_X^{(n)}(Y) = \delta_X^{(n_0)}(Y)$$

This operation is called **reconstruction** and is denoted by $\rho_x(Y)$. Formally,

$$\rho_x(Y) = \lim_{n \to \infty} \rho_X^{(n)}(Y) \tag{7}$$

In some applications it is desirable that one connected component of X is marked by several markers Y. If it is not acceptable for the sets grown from various markers to become connected, the notion of influence zones can be generalized to **geodesic influence zones** of the connected components of set Y inside X. Generalization of reconstruction to gray-scale images is from the fact that any increasing transformation defined for binary images can be extended to gray-level images. The generalization is achieved by viewing a gray-level image I as a stack of binary images obtained by successive thresholding–this is called threshold decomposition of image I. LetJ, I be two gray-scale images:w defined on the same domain D, $J_{(p)} \leq I_{(p)}$, the gray-scale reconstruction $\rho_I(J)$ of image I from image J is given by

$$\forall p \in D, \rho_I(J)(p) = max\{k \in [0, N], p \in \rho_{T_k}[T_k(J)]\} \tag{8}$$

Recall that binary reconstruction grows those connected components of the mask which are marked. The gray-scale reconstruction extracts peaks of the mask I that are marked by J.

Chapter 2

Segmentation

Image segmentation is one of the most important steps leading to the analysis of processed image data–its main goal is to divide an image into parts that have a strong correlation with objects or areas of the real world contained in the image. We may aim for **complete segmentation**, which results in a set of disjoint regions corresponding uniquely with objects in the input image, or for **partial segmentation**, in which regions may not correspond directly with image objects [26]. A complete segmentation of an image R is a finite set of regions $R_1,\ldots,R_S$,

$$R = \bigcup_{i=1}^{S} R_i \tag{1}$$

$$R_i \cap R_j = o \quad i \neq j \tag{2}$$

The segmentation is partial if either of conditions 1 and 2 above are not fulfilled.

To achieve a complete segmentation, cooperation with higher processing levels which use specific knowledge of the problem domain is necessary. However, there is a whole class of segmentation problems that can be solved successfully using lower-level processing only. In this case the image commonly consists of *contrasted* objects located on uniform background–simply assembly tasks, blood cells, printed characters, etc. Here, a simple global approach can be used and the complete segmentation of an image into objects and background can be obtained. Such processing is context independent; no object-related model is used, and no knowledge about expected segmentation results contributes to the final segmentation.

If partial segmentation is the goal, an image is divided into separate regions that are

homogeneous with respect to a chosen property such as brightness, color, reflectivity, texture, etc. If an image of a complex scene is processed, for example, an aerial photograph of an urban scene, or defective steel images, a set of possibly overlapping homogeneous regions may result. The partially segmented image must then be subjected to further processing, and the final image segmentation may be found with the help of higher-level information.

Totally correct and complete segmentation of complex scenes usually cannot be achieved in this processing phase, although substantial reduction in data volume offers an immediate gain. A reasonable aim we choose is to achieve partial segmentation which can be input to the next higher-level processing.

Image data ambiguity is one of main segmentation problems, often accompanied by information noise. Segmentation methods can be divided into three groups according to the dominant features they employ: First is **global knowledge** about the image or its part; the knowledge is usually represented by a histogram of image features. **Edge-based** segmentations form the second group, and **region-based** segmentations the third–many different characteristics may be used in edge detection or region growing, for example, brightness, texture, velocity field, etc. The second and the third groups solve a dual problem. Each region can be represented by its closed boundary, and each closed boundary describes a region. Because of the different natures of the various edge- and region-based algorithms, they may be expected to give somewhat different results and consequently different information. The segmentation results of these two approaches can therefore be combined in a single description structure. A common example of this is a region adjacency graph, in which regions are represented by nodes and graph arcs represent adjacency relations based on detected region borders. Generally, image segmentation systems follow the rules below.

- Regions of an image segmentation should be uniform and homogeneous with respect to some characteristics.
- Region interiors should be simple and without any small holes.
- Adjacent regions of a segmentation should have significantly different values with respect to the characteristics on which they are uniform.
- Boundaries of each segment should be simple, not ragged, and must be spatially accurate.

Following are some of the methods we investigated and presented are the results on various kinds of defects

2.1 Thresholding

Gray-level thresholding is the simplest segmentation process. Many objects or image regions are characterized by constant reflectivity or light absorption of their surfaces; a brightness constant or *threshold* can be determined to segment objects and background. thresholding is computationally inexpensive and fast–it is the oldest segmentation method and is still ideally used in simple applications; also it can easily be done in real time using specialized hardware.

Complete segmentation can result from thresholding in simple scenes. Thresholding is the transformation of an input image f to an output (segmented) binary image g as follows:

$$\begin{aligned} g(i,j) &= 1 \quad \text{for } f(i,j) \geq T \\ &= 0 \quad \text{for } f(i,j) < T \end{aligned}$$

where T is the threshold, $g(i,j) = 1$ for image elements of objects, and $g(i,j) = 0$ for image elements of the background (or vice versa).

2.1.1 Global Thresholding

Search all the pixels $f(i,j)$ of the image f. An image element $g(i,j)$ of the segmented image is an object pixel if $f(i,j) \geq T$, and is a background pixel otherwise.

Algorithm 1: Global Thresholding

If objects do not touch each other, and if their gray-level are clearly distinct from background gray-level, thresholding is a suitable segmentation method. Almost all the defects, which have protrusions like dents, splitters allow for segmentation through this method.

2.1.2 Adaptive thresholding

Only under very unusual circumstances, like an image taken at dark lighting conditions, where only dents appear as bright regions, in dark background, can basic thresholding

Set minrange;
Set window;
repeat (* for each pixel *)
find min and max under a selected window;
range = max-min;
if range > minrange
  then T = (max+min)/2;
else
  T=max-minrange/2;
until all pixels processed.

where T is the threshold for the selected window.

Algorithm 2: Adaptive Thresholding

be successful using a single threshold for the whole image (global thresholding). Since even in very simple images there are likely to be gray-level variations in objects and background; this variation may be due to non-uniform lighting, non-uniform input device parameters or a number of other factors. Segmentation using variable thresholds (also called *adaptive thresholding*), in which the threshold value varies over the image as a function of local image characteristics, can produce the solution in these cases [18].

A global threshold is determined from the whole image f :

$$T = T(f)$$

on the other hand, local threshold are position dependent:

$$T = T(f, f_c)$$

where f_c is that image part in which the threshold is determined. One option is to divide the image f into sub images f_c and determine a threshold independently in each sub image; then if a threshold cannot be determined in some sub image, it can be interpolated from thresholds determined in neighboring sub-images. Each sub-image is then processed with respect to its local threshold.

2.2 Edge-based segmentation

Edge-based segmentation represents a large group of methods based on information about edges in the image; it is one of the earliest segmentation approaches and still remains very

important. Edge-based segmentation rely on edges found in an image by edge detecting operators-these edge mark image locations of discontinuities in gray-level, color, texture, etc. A variety of edge detecting operators like gradient,Canny edge detector [6] etc. can be used for this purpose, but the image resulting from edge detection cannot be used as a segmentation result. Supplementary processing steps must follow to combine edges into edge chains that correspond better with borders in the image. The final aim is to reach at least a partial segmentation–that is , to group local edges into an image where only edge chains with correspondence to existing objects or image parts are present.

2.3 Region-based segmentation

The aim of the segmentation methods described in the previous section was to find borders between regions; the following methods construct regions directly. It is easy to construct regions from the borders, and it is easy to detect borders of existing regions. However, segmentations resulting from edge-based methods and region-growing methods are not usually exactly the same, and a combination of results may often be a good idea. Region growing techniques are generally better in noisy images, where borders are extremely difficult to detect and are a natural choice for Surface Inspection application. Homogeneity is an important property of regions and is used as the main segmentation criterion in region growing, whose basic idea is to divide an image into zones of maximum homogeneity. The criteria for homogeneity can be based on gray-level, color, texture, shape, model (using semantic information) [10] etc. Properties chosen to describe regions influence the form, complexity, and amount of prior information in the specific region-growing segmentation method.

2.4 Morphological Segmentation

Assuming that the image objects are connected regions of rather homogeneous intensity levels, one should be able to extract these regions by using some neighborhood properties rather than purely spectral properties as for histogram-based segmentation techniques. Indeed, a high gray-scale variation between two adjacent pixels may indicate that these two pixels belong to different objects.

The morphological approach to image segmentation combines region growing and

edge detection techniques [26]. It groups the image pixels around the regional minima of the image and boundaries of adjacent groupings are precisely located along the crest lines of the gradient image. This is achieved by a transformation called the *watershed transformation* [28].

2.5 Watershed transformation

Let us consider the topographic representation of a gray-level scale image. Now, let a drop of water fall on such a topographic surface. By gravity, it will flow down along to steepest slope path until it reaches a minimum [28]. The whole set of points of the surface whose steepest slope paths reach a given minimum constitutes the *catchment basin* associated with this minimum. The *watersheds* are the ridges dividing adjacent catchment basins. This is illustrated in Fig 2.1 Provided that the input image has been transformed so as

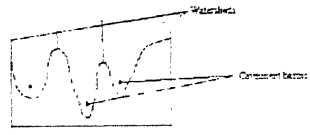


Figure 2.1: Minima, Catchment basins, and Watersheds on the topographic representation of a gray scale image. Fig shows simplified 1–D example of a Watershed segmentation– local minima of a gray–level (altitude) yield catchment basins, local maxima define the watershed lines.

to output an image whose minima mark relevant image objects and whose crest lines correspond to image object boundaries, the watershed transformation will partition the image into meaningful regions.

There are two basic approaches to watershed image segmentation [28]. The first one starts with finding a *downstream* path from each pixel of the image to a local minimum of image surface altitude. A catchment basin is then defined as the set of pixels for

which their respective downstream paths all end up in the same altitude minimum. While downstream paths are easy to determine for continuous altitude surfaces by calculating the local gradients, no rules exist to define the downstream paths uniquely for digital surfaces. The second watershed segmentation approach represented by Vincent and Soille in their seminal paper [28] makes the idea of real-time implementation practical. This approach is essentially dual to the first one: instead of identifying the downstream paths, the catchment basins fill from the bottom.

2.5.1 Definition in terms of flooding simulations

Consider the gray tone image as a topographic surface and assume that holes have been punched in each regional minima of the surface. The surface is then slowly immersed into a lake. Starting from the minima at the lowest altitude, the water will progressively flood the catchment basins of the image. In addition, dams are erected at the places where the water coming from two different minima would merge. At the end of this flooding procedure, each minimum is completely surrounded by dams, which delineate its associated catchment basin. The resulting dams correspond to the watersheds. They provide us with a partition of the input image into its different catchment basins.

Formally: I being the gray-scale image under study, denote h_{min} the smallest value taken by I. Similarly denote h_{max} the largest value taken by I. In the following $T_h(I)$ stands for the threshold of I at level h.

$$T_h(I) = \{p \in D_I, I(p) \leq h\} \tag{3}$$

To simulate the immersion procedure described above, we *start* from the set $T_{h_{min}}(I)$, the points of which being those first reached by the water. These points constitute the starting set of our recursion. Thus,

$$X_{h_{min}} = T_{h_{min}}(I) \tag{4}$$

$X_{h_{min}}$ is made of the points of I which belong to the minima of lowest altitude. Let us now consider the threshold of I at level $h_{min} + 1$, i.e., $T_{h_{min}+1}(I)$. Obviously, $X_{h_{min}} \subseteq T_{h_{min}+1}(I)$. Now, Y being one of the connected components of $T_{h_{min}+1}(I)$, there are three possible relations of inclusion between Y and $Y \cap X_{h_{min}}$.

1. $Y \cap X_{h_{min}} = \emptyset$: in this case, Y is a new minimum of I. All the surrounding pixels do not belong to $T_{h_{min}}(I)$ and have therefore a gray level strictly greater

Figure 2.2: The three possible inclusion relations between Y and $Y \cup X_{h_{min}}$

than $h_{min} + 1$. The minimum thus discovered is "pierced", hence its corresponding catchment basin will be progressively filled up with water.

2. $Y \cap X_{h_{min}} \neq \emptyset$ and is connected: in this case, Y corresponds exactly to the pixels belonging to the catchment basin associated with the minimum $Y \cap X_{h_{min}}$ and having a gray level lower or equal to $h_{min} + 1$.

3. $Y \cap X_{h_{min}} \neq \emptyset$ and is not connected: we therefore notice that Y contains different minima of I. Denote $Z_1, Z_2, \ldots, Z_k$ these minima, and let Z_i be one of them. At this point, the best possible choice for $C_{h_{min-1}}$ is given by the geodesic influence zone of Z_i inside Y ($IZ_{Z_i}(Y)$).

These inclusion relationships are illustrated in Fig 2.2. the *second* set of recursion is as follows:

$$X_{h_{min}} = \min h_{min} + 1 \cup IZ_{T_{h_{min}-1}(I)}(X_{h_{min}}) \tag{5}$$

This relation holds of course for all levels h. The set of the **catchment basins** of the gray-scale image I is equal to the set $X_{h_{min}}$ obtained after the following recursion:

1. $X_{h_{min}} = T_{h_{min}}(I)$,

2. $\forall h \in [h_{min}, h_{max} - 1], X_{h-1} = min_{h+1} \cup IZ_{T_{h-1}(I)}(X_h)$.

The recursion relation between two successive levels is illustrated in Fig 2.3.

To speed up the computations, one has to design algorithms taking into account the fact that, at a given step, only the values of a small number of pixels may be modified. Rather than scanning the entire image to modify only two pixels, the algorithm should be designed to have a direct access to these pixels. Therefore, in the following, we suppose that the image pixels are stored in a simple array, and that the following two conditions are satisfied:

1. *Random access* to the pixels of an image.

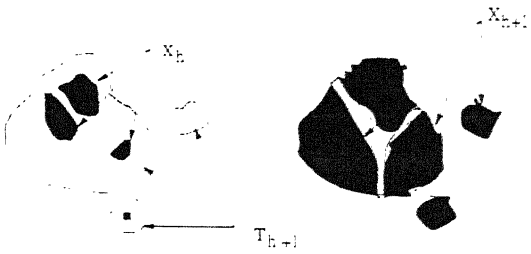


Figure 2.3: Recursion relation between X_h and X_{h-1}

2. *Direct access to the neighbors* of a given pixel (its 4 neighbors in 4-connectivity, 6 on a hexagonal grid, 8 on 8-connectivity, etc.).

If these two prerequisites are fulfilled, one is able to design extremely efficient morphological algorithms. The immersion simulation based algorithm has to successively threshold the image under study, and compute geodesic influence zones of one threshold inside the next one as fast as possible. In the sequel, for the sake of clarity, this algorithm is explained in two steps. In order to have a direct access to the pixels at a given level, the first step consists in an initial *sorting* of the pixels in the increasing order of their gray values. The method described in section 2.5.2 completes this very efficiently, since it exploits the particular structure of our data. It runs in linear time with respect to the number of pixels to be sorted. In the second step a fast computation of geodesic influence zones is enabled by a breadth-first scanning of each threshold level. This particular scanning is implemented via the use of a *queue* (first-in-first-out data structure). Notice that many morphological transformations can be efficiently performed by algorithms based on queue structure. This second step, called the *flooding* step, is detailed in section 2.5.2.

2.5.2 The Sorting Step

During the sorting step, a brightness histogram is computed. Simultaneously a list of points to pixel of gray-level is created and associated with each histogram gray-level to enable direct access to all pixels of any gray-level using a radix sorting [17]. Information about the image pixel sorting is used extensively in the flooding step.

2.5.3 The Flooding Step

Once the pixels have been sorted, we proceed to the progressive flooding of the catchment basins of the image. Suppose the flooding has been done up to a given level h and underlying is hexagonal grid. Every catchment basin already discovered–i.e., every catchment basin whose corresponding minimum has an altitude lower or equal to h –is supposed to have a unique label. Thanks to the initial sorting, we now access the pixels of altitude $h+1$ directly Those pixels among them which have an already labeled pixels as one of their neighbors are put into the queue. Starting from these pixels, the queues structure enables to extend the labeled catchment basins inside the mask of pixels having value MASK, by computing geodesic influence zones. After this step, only the *minima* at level $h+1$ have not been reached. Since, they are not connected to any of the already labeled catchment basins. Therefore a second scanning of the pixels at level $h+1$ is necessary to detect the pixels which still have value MASK, and to give a new label to the thus discovered catchment basins.

Not only does the use of a queue of pixels speed up the computations, it also allows to solve the accuracy problems encounter by most of the non-immersion based algorithms. Now, in order to get perfectly located watershed arcs, the successive geodesic SKIZ involved i the process have to be as good as possible. The first thing to notice is that, according to the discrete distance associated with the underlying grid, the components may well not be a line, but a very thick area (Recall that the distance between two pixels is equal to the minimal number of grid edges to cross to go from one to the other). The plateaus at elevation h are currently being flooded and let p be the current pixel. A simplistic rule would be to say that p is necessarily a watershed-pixel (i.e., belongs to the set of watershed lines) if it has a watershed-pixel in its neighborhood. Similarly, declaring that any pixel which has two pixels with different labels in its neighborhood is a watershed-pixel may result in deviated watershed lines. Let us stress that these problems are not specific of the hexagonal grid and also exist with square ones.

To get rid of such difficulties, one may think of resorting to better discrete distance in the geodesic SKIZ computations. It is even possible to make use of actual Euclidean distances by adapting some algorithms to the present case. However, this would involve propagating *vectors* rather than distances and would put a considerable burden on the entire flooding step. Therefore, these ideas have not been retained in the present implementations.

Instead this algorithm restrict to the distance induced by the used discrete grid. The

idea is to make use of a work image where the successive geodesic distances are actually stored during the breadth-first propagation. In conjunction with carefully written rules for the propagation of the labels inside the plateaus, this results in very well located watershed lines, even in the case of minima embedded in large plateaus. Note that the algorithm is designed to yield a tessellation of the image in its different catchment basins. Only the pixels which are exactly "half-way between" two catchment basins are given a special value.

At this point, to get rid of the WSHED-pixels (i.e., to obtain a real tessellation of the image in its different catchment basins), it suffices to give them the value of one their labeled neighbors (in fact, the pixels belonging to thick watershed areas must be processed differently). On the other hand, if we want to separate the different catchment basins, it suffices to give value WSHED to the labeled pixels having in their neighborhood at least one pixel with a *smaller* label.

2.6 Multiscale Gradient

Image segmentation is an essential step for many image analysis tasks, such as object recognition. the goal of image segmentation is to partition an image into homogeneous regions and locate the contours of the regions as accurately as possible. A large number of techniques and algorithms have been proposed for image segmentation. Among them, those based on watershed transformation can potentially provide accurate segmentation with very low computational cost.

For image segmentation, watershed transformation starts with the gradient of the image to be segmented. It views the gradient image as a three-dimensional (3-D) surface where gradient values act as surface heights. Intensity edges in the image to be segmented generally have high gradient values which appear as **watershed lines** (also known as mountain ridges) on the 3-D surface. The watershed lines partition the gradient image into different catchment basins on the 3-D surface. The watershed lines partition the gradient image into different catchment basins which correspond to homogeneous regions of the image to be segmented. Watershed transformation involves a search for watershed lines in the gradient image. Therefore, the performance of a watershed-based image segmentation method depends largely on the algorithm used to compute the gradient.

Conventional gradient algorithms exhibit a serious weakness for watershed-based image segmentation. A conventional gradient operator, such as the first partial derivative of

Gaussian filter and morphological gradient operators, produces too many local minima because of noise and quantization error within homogeneous regions. Each minimum of the gradient introduces a catchment basin with the watershed transformation. Hence, these gradient operators result in over-segmentation, e.g. a homogeneous region may be partitioned into a large number of regions and proper contours are lost in a multitude of false ones. A straight forward method to deal with this problem is to threshold the gradient. However, conventional gradient operators produce low gradient values at blurred edges, even though the intensity change between the two sides of an edge may be high. By thresholding, one cannot eliminate the local minima caused by noise and quantization error while preserving those produced by blurred edges. Another solution for this problem is to extract *markers* and impose them on the gradient image, which may require prior information about the objects and background to be segmented. After watershed transformation, region merging or relaxation labeling is usually performed to further remove false contours. This process may be much more computationally expensive than watershed transformation, because too many catchment basins have to be merged. This greatly decreases the entire segmentation. In this chapter we discuss the multiscale gradient algorithm [30]. This algorithm efficiently enhances blurred edges while being very robust to multi-edge interactions. This enhancement increases the gradient value for blurred edges above those caused by noise and quantization error. Also we present an algorithm to eliminate the local minima produced by noise quantization error [30]. Finally, we show how to exploit the multiscale gradient algorithm for region merging.

2.6.1 The Multiscale gradient algorithm

Many gradient operators and edge detection algorithms have been based on the step edge model [6]. However, ideal step edges do not exist in natural images since every edge is blurred to some extent. A blurred edge can be modeled by a ramp and the intensity change between two sides of the edge is referred to as *edge height*. For a ramp edge, the output of a conventional gradient operator, is the slope of the edge. Hence, the ramp edge cannot be separated from noise and quantization error by thresholding if the slope of the edge is small. Fig 2.4 shows a 1-D example where a ramp edge and step edge are digitized in order to illustrate the effect of quantization error. The ideal gradient operator for watershed transformation is the one whose output is equal to the input edge height, but not the edge slope. Mono-scale morphological gradient operator performance depends on

the size of structuring element B. If B is large, the output of this gradient operator for a ramp edge is equal to the edge height. Unfortunately, large structuring elements result in serious interaction among edges which may lead to gradient maxima not coinciding with edges. However, if the structuring element is very small, this gradient operator has a high spatial resolution, but produces a low output value for ramp edges.

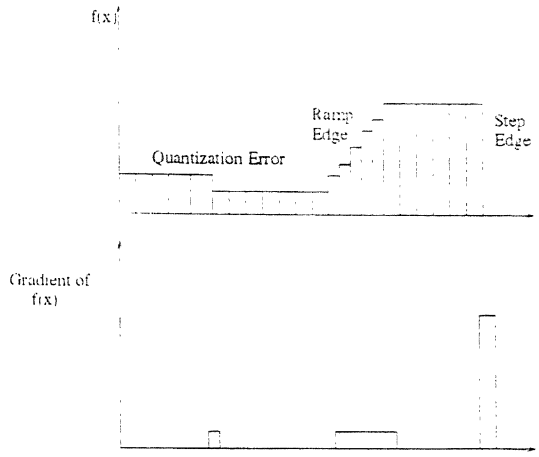


Figure 2.4: The Conventional Gradient Operator

In order to exploit the advantages of both small and large structuring elements, we propose a multiscale morphological gradient algorithm. Let B_i, for $0 \leq i \leq n$, denote a group of square structuring elements. The size of B_i is $(2i+1) \times (2i+1)$ pixels, i.e. B_o contains only one pixel and B_i is a 3×3 square and so on. The multiscale gradient of a function(f) is defined by

$$MG(f) = \frac{1}{n} \sum_{i=1}^{n} [((f \ominus B_i) - (f \ominus B_i) \oplus B_{i-1})] \tag{6}$$

For a step edge, the operation $((f \oplus B_i) - (f \ominus B_i)) \ominus B_{i-1}$ produces a line of two pixels wide which coincides with the edge. The intensity (height) of the line is equal to the edge height. Hence, the multiscale gradient algorithm is equivalent to the mono-scale morphological gradient operator in this case. In practice, it is more robust to noise due to the averaging operation used in the algorithm.

For a ramp edge we denote respectively the edge width and height by w and h. The operation $((f \oplus B_i) - (f \ominus B_i)) \ominus B_{i-1}$ produces a line coinciding with the edge. The cross section of the line appears as a trapezoid if $i < (w+2)/4$ and as a triangle otherwise.

The width of the bottom side of the trapezoids or triangles is always equal to $w-2$ pixels. the height of the trapezoid is $2ih/w$ and that of the triangle is $h(w+2)/(2w)$, which are greater than the edge slope h/w. The value of MG(f) approaches to $h(w+2)/(2w)$ if n is large enough. Therefore, the multiscale algorithm responds effectively to ramp edges without enlarging edges. The multiscale gradient algorithm is very robust to edge interaction. The location of gradient maxima corresponding to one edge is not disturbed by the presence of other edges. The distance between two adjacent edges is denoted by d. When $i < d/2$, $((f \ominus B_i) - (f \ominus B_i)) \ominus B_{i-1}$ correctly produces gradient maxima which coincide with the edges. While $i \geq d/2$, $((f \ominus B_i) - (f \ominus B_i)) \ominus B_{i-1}$ fills the gap between the two adjacent maxima with the value of the smaller maximum. By averaging the values produced by $((f \ominus B_i) - (f \ominus B_i)) \ominus B_{i-1}$ for all i, MG(f) keeps the gradient maxima at the correct locations. For the interaction between ramp edges, the situation is similar. It should be noted that the distance d between edges must be at least three pixels since B_i is a 3×3 square. If the distance is smaller than three pixels, the image should be up sampled before computing gradient. The structuring element B_i in equation 6 could be of any shape satisfying the relation $B_0 \subseteq B_1 \subseteq B_2 \ldots \subseteq B_n$. We use a group of square structuring elements because of its low computational cost. Moreover, one may use several groups of directional structuring elements, e.g. line segments, for specific applications. The gradients of different directions are computed with the corresponding groups of structuring elements and the maximum value of these gradients is taken as the final result. According to our experiments, this approach produces more local minima than using square structuring elements.

2.6.2 Elimination of small local minima

Small local minima is defined as local minima consisting of a small number of pixels or having a low contrast with their neighbors [30]. This kind of local minima in gradient images is generally caused by noise or quantization error, and therefore should be eliminated.

Local minima consisting of a small number of pixels are eliminated by dilation with a square structuring element B_s of 2×2 pixels, denoted by MG(f)$\ominus B_s$. To remove the local minima with a low contrast a constant denoted by h is first added to the dilated gradient image. Then the local minima with a contrast lower than h can be filled using the reconstruction by erosion of MG(f) from (MG(f)) $\ominus B_s + h$. Hence, the final gradient

image can be expressed as $\phi^{(rec)}[MG(f) \ominus B_s + h, MG(f)]$.

This algorithm is illustrated in Fig 2.5, where MG(f) has six local minima. Local minima (LM) 3 and 5 consisting of one pixel are removed by the dilation, while local minima 2,4 and 6 having a contrast lower than h are eliminated by the reconstruction by erosion. Local minimum 1 is not removed because it is both wide and deep. The constant h is used to control the number of segmentation regions. As h increases, the number of regions produced decreases. The reconstruction by erosion fills all of the minima where the contrast is lower than h, irrespective of their absolute values. However, thresholding removes only the minima with low absolute values. For the MG(f) shown in Fig 2.5, thresholding cannot remove local minima 2 and 3.

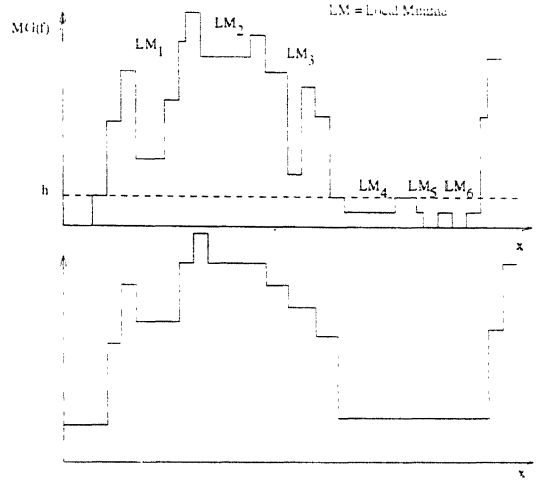


Figure 2.5: Elimination of small local minima.

2.7 Watersheds in Image segmentation

The watershed transformation constitutes one of the most powerful segmentation tools provided by mathematical morphology [26]. In this section, the worked segmentation addresses the extraction of the different objects present in the image under study. As concerns the binary case, this comes down to the separation of the partially overlapping objects provided there are no artifacts present. Its solution is based on a *marking* of the different components that are to be segmented. A *marking function* is then constructed, whose different catchment basins correspond to the desired objects. Here, the marking

Figure 2.6: Original Pinch mark image

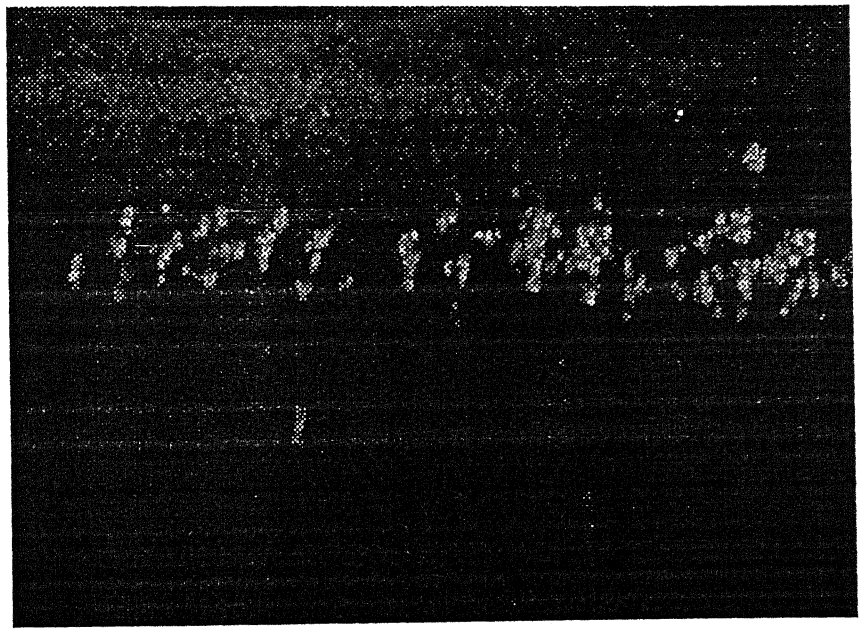


Figure 2.7: Multiscale gradient with h = 10 and number of iterations = 1

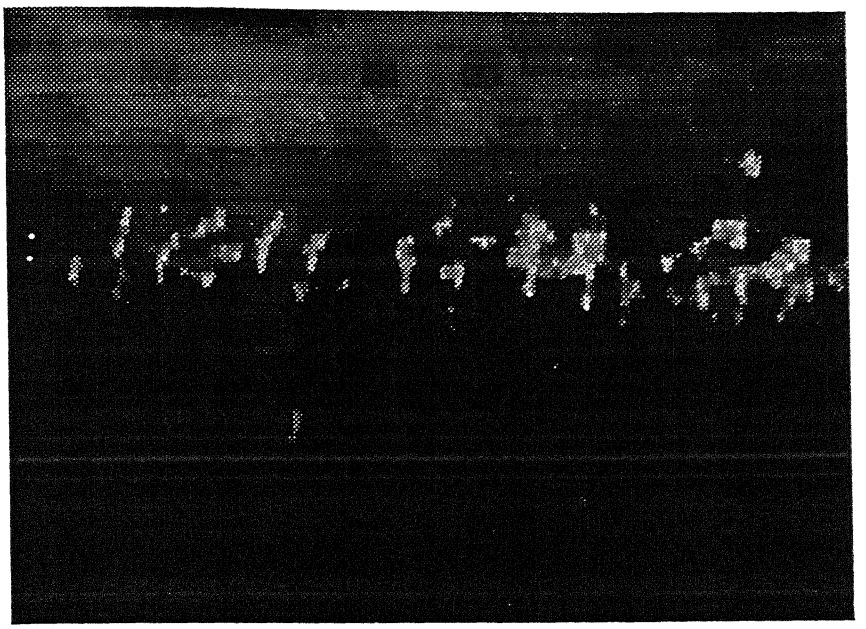


Figure 2.8: Multiscale gradient with h = 20 and number of iterations = 8

Figure 2.9: Watershed with 2.7 and 2.8 as marker after adaptively thresholding between 0 and 100

function is nothing but the opposite of the *distance function* of our binary image, i.e., the function which associates with every feature point the opposite of its distance to the background.

As concerns gray-scale images, segmenting them means dividing them into regions: generally, one of them stands for the background whereas each of the others corresponds to one of the objects or areas to be extracted. This segmentation comes down to the extraction of the contours of the desired objects. Now, the problem is to clearly define what is a contour and what is not. Some well-known methods resort to the zero-crossings of the second derivative of the function representing the image I under study. Other edge detectors can be computationally adapted to arbitrary edges profiles. In the field of Mathematical morphology, another kind of approach is commonly used: the starting point is to say that the contours of an image correspond to lines where the gray-tone is varying quickly compared to the neighborhood. Suppose now that we have determined an image grad(I) where the value of each pixel corresponds to the modulus of the gradient at this point(in the following, this image is referred to as a *gradient image).* If we regard it as a relief, the searched contours correspond to some *crest-lines* of this function. At this point, one can consider using on grad(I) the *gray-scale skeleton* as crest-line detector. the problem with this transformation is that it extracts *all* the crest-lines of a given image. This is not what is expected, since the contours that should be extracted are closed ones. Therefore, one has to remove parasitic dendrites of the skeleton, i.e., to resort to the watershed transformation. According to the gradient which is being used, we define the contours of image I as the *watersheds* of *gradient.*

The watersheds of the gradient are thus at the basis of the general morphological approach to segmentation that we briefly present now. The image which will be used to illustrate/ the present segmentation methodology is denoted I and is displayed in Fig 2.10. It is a 640×480 image of a pinch mark, from which the blobs of marks to be extracted. This is a rather difficult problem, since on the one hand, the noise level is relatively high and on the other hand, the desired marks do not have a fixed size and orientation, and can be mistaken for other features. Simple methods based on thresholding or top-hats do not work, so that one has to make use of more advanced tools.

In fact the brutal computation of watersheds of the gradient does not constitute a good segmentation method either. Indeed, whatever gradient is used, the simple computation of its watersheds mostly results in an *over-segmentation* i.e., the correct contours are lost in a mass of irrelevant ones. This is true even if one had taken the precaution of

filtering the initial image or its gradient. For example, I' (see Fig 2.10(b)) was obtained by performing on I an alternating sequential filtering. A morphological gradient of I' was then determined. This gradient is displayed in Fig. 2.10(c). Now in Fig. 2.10(d), one can see the watersheds of this gradient. In many cases, the over-segmentation is simply due to noise. But, as in the present example, some irrelevant arcs may also correspond to objects that do not have to be extracted, and whose contours should not appear in the final segmented image. In both cases so as to get rid of this over-segmentation, one has two possibilities:

- Remove the irrelevant contours elements.
- Modify the gradient function so that the resulting catchment basins only correspond to the desired objects.

The first choice is the most natural and classical one: on the other hand, the watershed image can be regarded as an image of contours, some of which having to be suppressed. On the other hand, one may consider the different catchment basins as regions and merge adjacent regions according to some similar criteria. These methods are referred to in literature as *region-growing algorithms*. A morphological region growing algorithm relying on watersheds is applied.

The second choice makes use of some external knowledge on the collection of images under study, in the sense that it requires and initial *marking*step. One has to use the knowledge available on the problem–shape of the desired objects, noise present on the image, darkness of the background, etc.–to design a robust algorithm for extracting markers of the different regions to be segmented [28]. By marker of a region, we mean a connected set of pixels (or even one single pixel) included in this region. On the example of 2.10, using again a topographic analogy, one can see that the pinch marks having a well defined shape. At this point, the idea is to detect markers of all blobs. Hence, a morphological multiscale gradient transformation is used as a marker. The result it yields is then skeletonized.

Once these markers are extracted, a morphological transformation based on gray-scale geodesic operations allows us to:

1. impose them as minima of a gradient function grad(I),
2. suppress all the other gradient minima (the insignificant ones) by filling up their catchment basins,

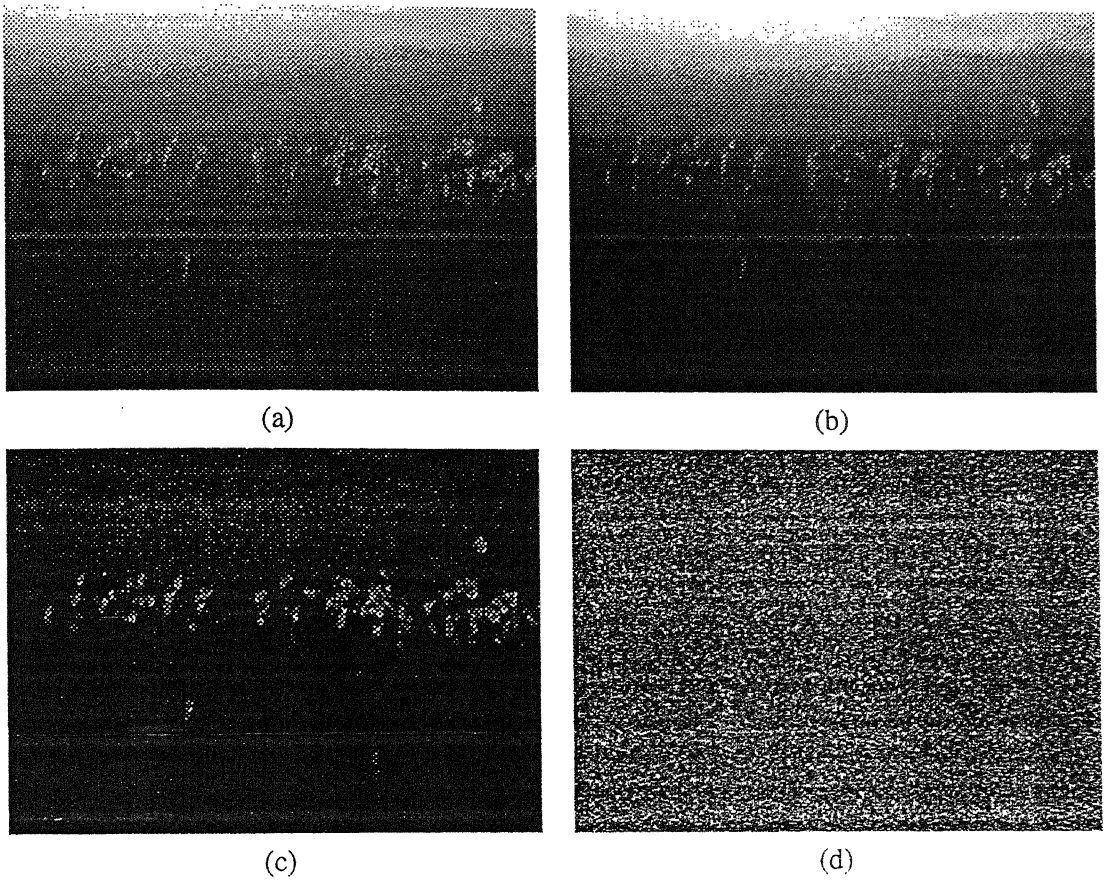


Figure 2.10: (a)*Original pinch mark image* (b)*After alternative sequential filtering* (c)*Morphological gradient of (b)* (d)*Watershed transformation of (c) resulted a over–segmentation*

3. preserve the most important crest-lines of grad(I) located between the markers.

This transformation is called *modification of the gradient homotopy* [28], or simply gradient modification. In some cases, the initial gradient has to be carefully chosen, since its most important crest-lines separating the extracted markers must be properly located. For the present application, the morphological gradient is sufficient. The computation of the watersheds of this modified gradient provides then the desired segmentation: the catchment basin of the background marker stands background itself whereas the boundaries of all other catchment basins–i.e., the watersheds–correspond to the desired object. The resulting contours are superimposed on the initial image. Finally, to extract the actual blobs from the thus contoured object, one has to use the already mentioned shape information: the blobs are the only objects whose skeleton is smooth etc., and not too long. The final segmentation is displayed in Fig. The methodology presented in this section is applied for various kinds of defects and suitable markers are observed.

2.8 Performance Evaluation

2.8.1 Groundtruth

In the early years of satellite imagery there was a need for obtaining the "correct" interpretation of the images. This was typically done by inspectors visiting the regions of the image (ground) based on which the content interpretation or *groundtruth* was defined. Today groundtruth is used for various image processing and analysis applications for refering interpretation. Groundtruth information should include

1. Location
2. Characterization (What defect?)
3. Severity

and is manually genrated typically by domain experts in this area. This information is compared with that of result obtained from the segmentation algorithm for finding the Error Index (See 2.8.2). We have collected a digital data set of wide class of defective steel images taken at different lighting conditions and different resolutions. Apparently they all have different levels of noise too. Also collected is the groundtruth of some of the images and groundtruth of pinch mark image is shown in Fig 2.11.

Figure 2.11: Groundtruth of Pinch Mark

To evaluate the performance of any segmentation algorithm we need (*i*) datasets of images and (*ii*) the corresponding symbolic groundtruth. Obtaining manually generated groundtruth, as is generally done, is subjective, labour-intensive, prohibitively expensive, and prone to errors. But we don't find any alternative for this at this moment. Unlike in OCR there is no similarity of content in the data set as steel images are wide varying in lighting conditions and defect characterizations(e.g. geometry, gauging etc).

Characterizing the performance of segmentation algorithms is important for many reasons:

1. *Performance prediction*: Typically segmentation is part of a bigger system, e.g., a Model Based Recognition system(MBR) or Content Based Image Retrieval(CBIR). Since the overall performance depends on the performances of the individual subsystems, the overall performance of the corresponding MBR/CBIR system is a function of the segmentation. Knowledge of end-to-end performance as a function of segmentation will allow us to predict the minimum level of segmentation required for achieving a specified overall MBR/CBIR system performance.

2. *Monitor progress*: In order to monitor progress in research of MBR systems defect

detection in steel industry, we need quantitative measures.

3. *Provide scientific explanations*: Understand the contribution to the accuracy improvement by various preprocessing image processing routines.

There are no perfect benchmarks for evaluating the segmentation quantitatively. Obviously, as the "required content" in a given image varies from application to application, the idea of generalized evaluation doesn't make sense. For evaluation of our segmentation, we choose polygon approximation for both segmented blobs and groundtruth(required region). We started with rectangular enclosing and the results are given for the same. Although rectangular approximation is coarse and prone to error, due to the enclosing of erroneous area at the corners, it is taken because the study is relative comparison of one method over the other and is taken as safe for experimentation.

2.8.2 Error Index

For evaluating the performance of segmentation one should have a quantitative measure. This metric should reflect the match/deviation from that of expected result with the human identifier. Here the groundtruth itself is not very accurate due to varied interpretation and conceptualization of defect. We formulated an Error Index (EI) for quantitatively evaluating the success of segmentation. This in turn would be useful for tuning the segmentation process.

Let G be a bounding polygon of groundtruth and M be the result of segmentation. Then EI is defined as

$$EI = \frac{||G - M|| + ||M - G||}{||M \cup G||} \tag{7}$$

where $-$ and $\cup$ are boolean difference and union respectively, gives a metric of percentage deviation of segmentation from the expected (groundtruth).

Chapter 3

Applications to Steel Defect Segmentation

3.1 Pinch Marks

Pinch marks are one of the challenging defects to segment. One of this kind is shown in Fig 2.10. Pinch marks are small local erosions on steel sheet. Typically they will be identified by the fact that number of blobs/marks per unit will be high. The marks even won't have sharp and well defined edges. Added to this complexity is noise present at the top of the image. Here one can clearly observe that simple operations like thresholding, top-hat doesn't work which are based on intensity variation. We applied watershed transformation on this with multi-scale gradient as marker. The marker was multiscale-gradient of original image with h value of 20 and with number of iterations as 20. This is shown in Fig 2.8. The watershed transformation on mono-scale gradient of the image (I) is shown in Fig 3.2. The resulting segmentation of watershed transformation on single scale gradient is a over–segmentation, which essentially doesn't represent the real world defects. For improving the segmentation we applied the same watershed transformation on a multiscale gradient shown in Fig 2.8 and the resulting segmentation is shown in Fig 3.3. The same result superimposed on original image is shown in Fig 3.4. The rectangle bounded result from segmentation is shown in Fig 3.5, which is suitable for performance evaluation comparing with groundtruth 2.8.

The groundtruthed image for the same defect is shown in Fig 2.11. This is a manually groundtruthed image. The groundtruth (expected) and segmentation result is shown

in Fig 3.6. The white bounding boxes are of groundtruth and red ones are a result of segmentation.

Figure 3.1: Pinch marks

3.2 Holes

Holes are one of the simplest to segment. One of this kind is shown in Fig. 3.7 They will be having sharp edges which lend themselves for good complete segmentation. A simple global thresholding does a partial segmentation but watershed based segmentation results a complete segmentation. The marker has been extracted by thresholding the original image with an experimented value of 125 and followed by closing operation for elimination of darker connected regions. Watershed transformation was performed on morphological gradient of the image. This is shown in Fig. 3.9.

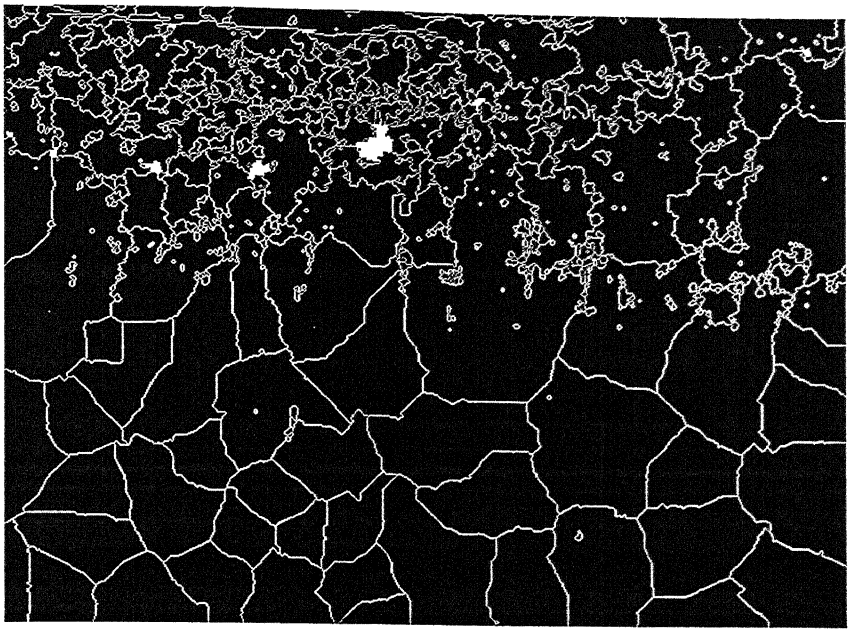


Figure 3.2: Watershed transformation on single-scale gradient resulting a over–segmentation

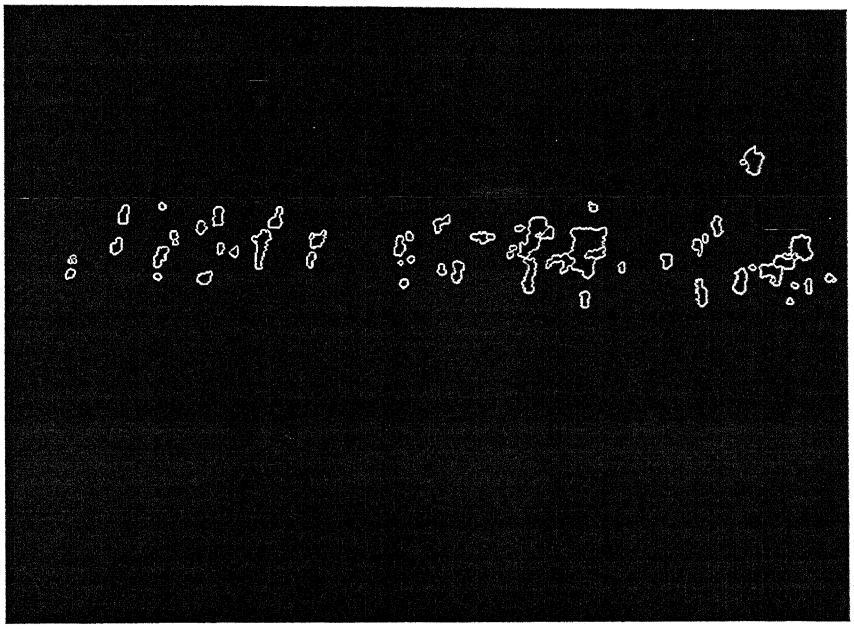


Figure 3.3: Watershed transformation on multiscale gradient resulting a comparable segmentation

Figure 3.4: Watershed transformation resulting on a multiscale gradient superimposed on original image demonstrates resulting of sensible segmentation

Figure 3.5: Rectangle bounded segmentation result

3.3 Rust Marks

Rust mark is shown in Fig 3.12. More description about their properties and occurring can be found in [21]. Morphologically, they can be described as segregation of brighter connected regions with a very small inter-blob distances. They can be thresholding by thresholding the multiscale gradient with a value of 40 on the whole image is quite dark. The final segmented image overlaid on groundtruth is shown in Fig 3.13.

3.4 Black Patch

Black patches are dark areas spread on the sheet as bigger blobs and are shown in Fig. 3.15.

Black patches can be considered as deep and large valleys in topological view and naturally extracting the regional minima as markers would be intuitive for watershed segmentation.

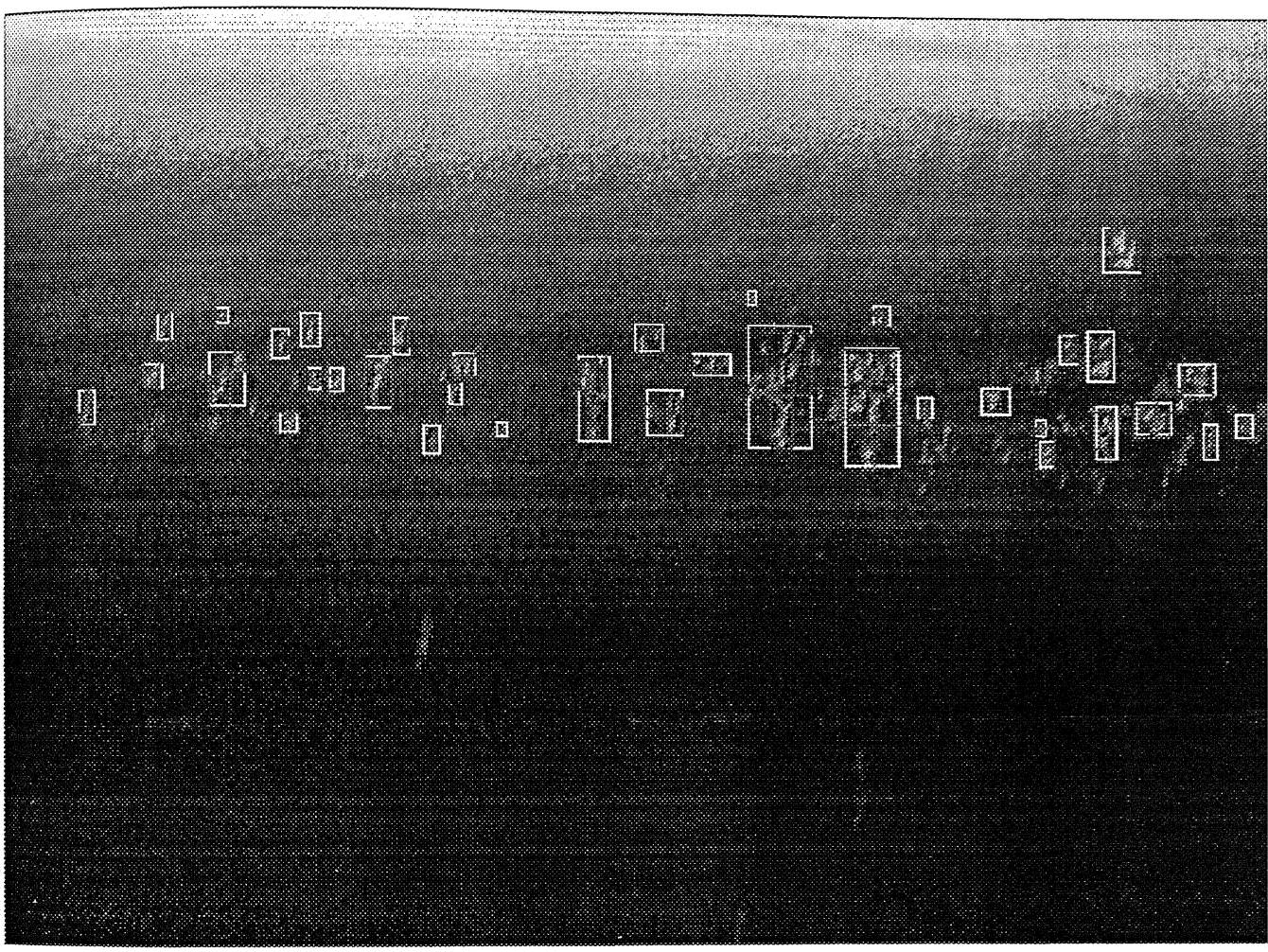


Figure 3.6: Result of Segmentation and Groundtruth superimposed

Figure 3.7: Hole

Figure 3.8: Gradient of the original image

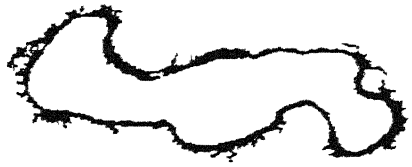


Figure 3.9: The marker used for watershed transformation

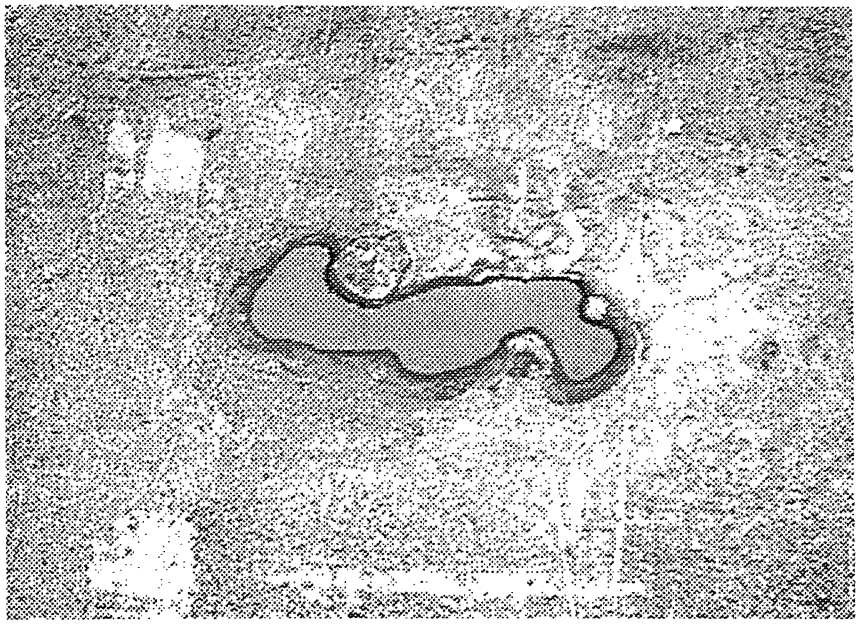


Figure 3.10: Watershed segmentation of hole

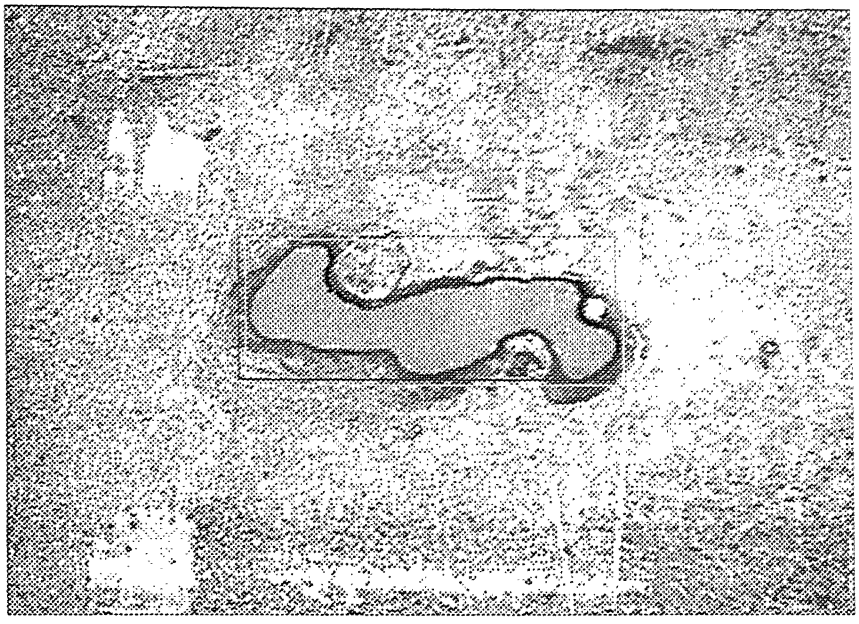


Figure 3.11: Result of segmentation superimposed on Groundtruth

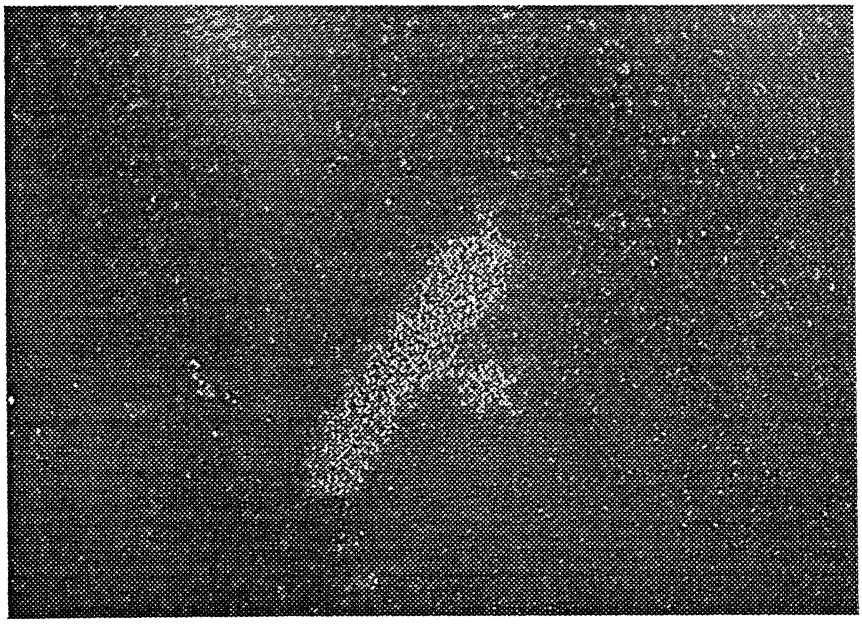


Figure 3.12: Segmentation of Rust mark

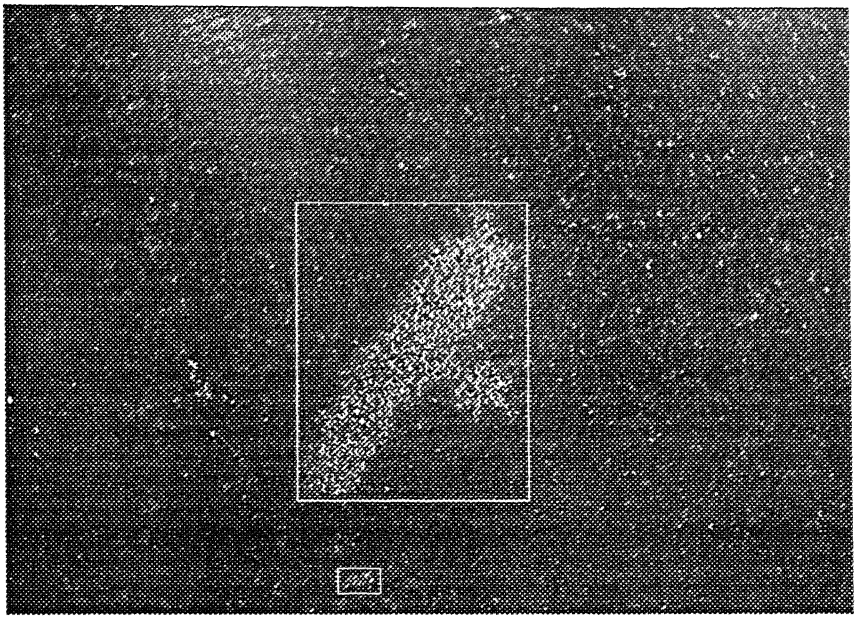


Figure 3.13: Segmentation of Rust mark superimposed with groundtruth

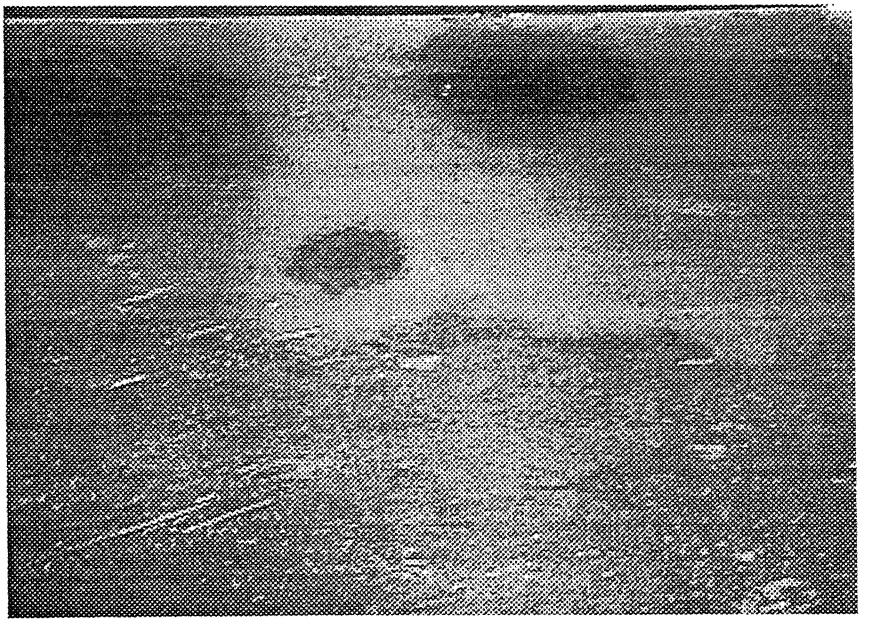


Figure 3.14: Black patches resulting from improper annealing

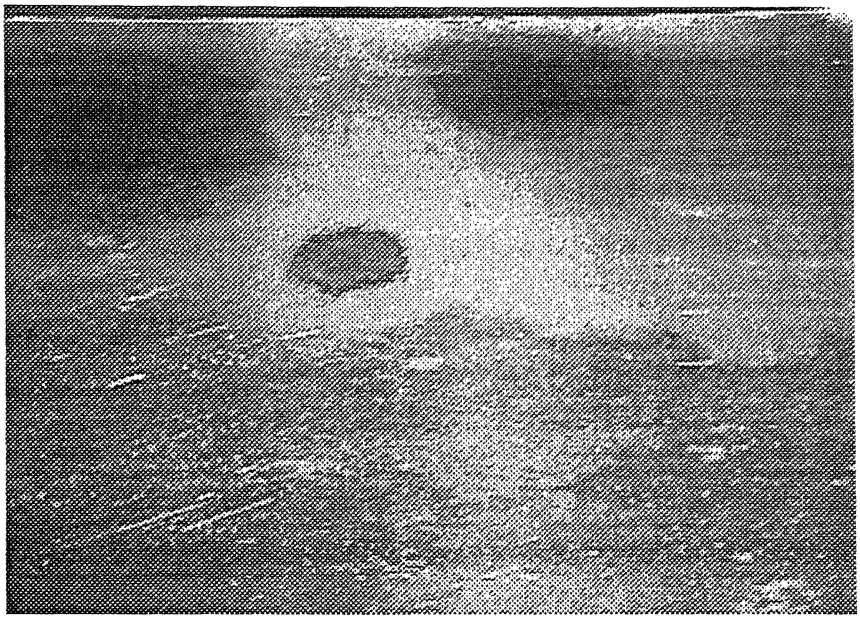


Figure 3.15: Segmentation of Pinch marks

3.5 Parameters and Computational Complexity of the algorithm

The first step in the overall algorithm is *pre-processing*. This is essential for the images having a lot of noise and those which were digitized under non-uniform lighting condition. We chose a low-pass filter based on morphological operations, described as Alternative sequential filter (See Section 1.9.4). Parameters involved in this step are kind of structuring element (k) and its dimensions ($a \times b$ for box type, a for cross type), and number of sequences of closing and opening (r). Typical values of r are

1. 0 for no or less noise
2. 2 for medium noise
3. 4 for high noise

The second step is finding the *multiscale gradient* [30] of the given image. As is said in above sections, this is being used instead of single scale gradient in order to suppress some unwanted local minimas for obtaining a sensible segmentation through watershed

transformation. Various parameters involved in this step are structuring element, h being threshold for multiscale gradient and n, the maximum scale. The maximum scale needs to be taken until the stability reaches in reconstruction in which case its computationally unaffordable. Hence we choose pragmatic values for this parameter and maximum value used was in the order of 15. The computational complexity is of cubic order and is given as

$$O\left(n^3 \quad |I|\right) \tag{1}$$

where $|I|$ represents the size of the image.

The last step in the segmentation is fast *watershed transformation* using immersion simulation [28]. The only parameter is structuring element. The computational complexity arises due to two steps–sorting and flooding simulation. For sorting we used radix sorting [17] and the computational complexity is linear time of image size $O\left(|I|\right)$ and the flooding step is of the order $(ab \quad |I|)$ and the overall complexity is $O\left(n^3 \quad |I|\right)$ approximately as $ab << n^3$.

3.5.1 Using performance evaluation to tune parameters

A process of tuning the parameters for the above multi-scale watershed process is illustrated in the Figures 3.16, 3.17 and 3.18. As can be clearly seen, the segmentation is very high in all of these images, and consequently will result in very high errors.

The evaluation function provides a tool for searching in the parameter space for better matches. However, this has not been done autonomously in this work. The final sensible segmentation is shown in Fig 3.3

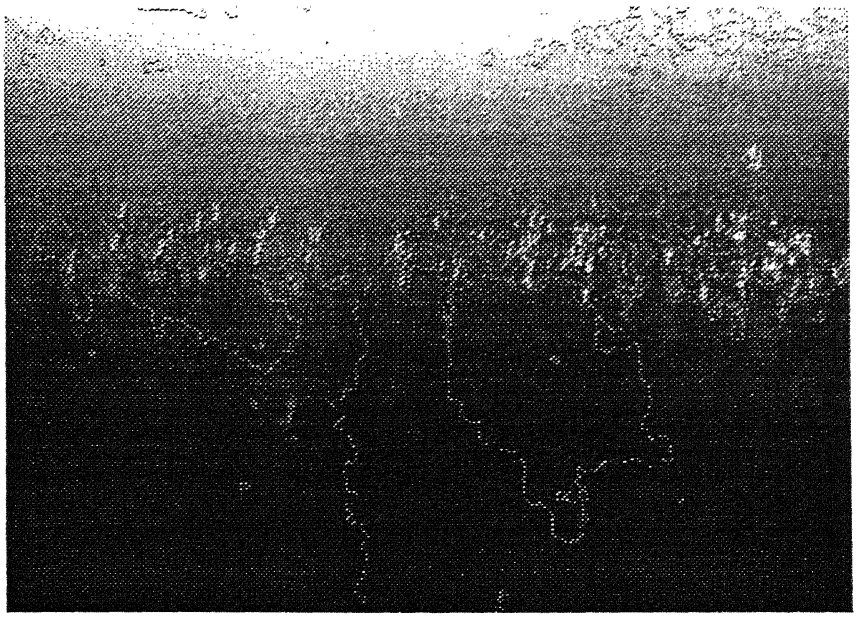


Figure 3.16: Watershed Segmentation with $h = 10$ and $n = 2$

Figure 3.17: Watershed Segmentation with $h = 10$ and $n = 9$

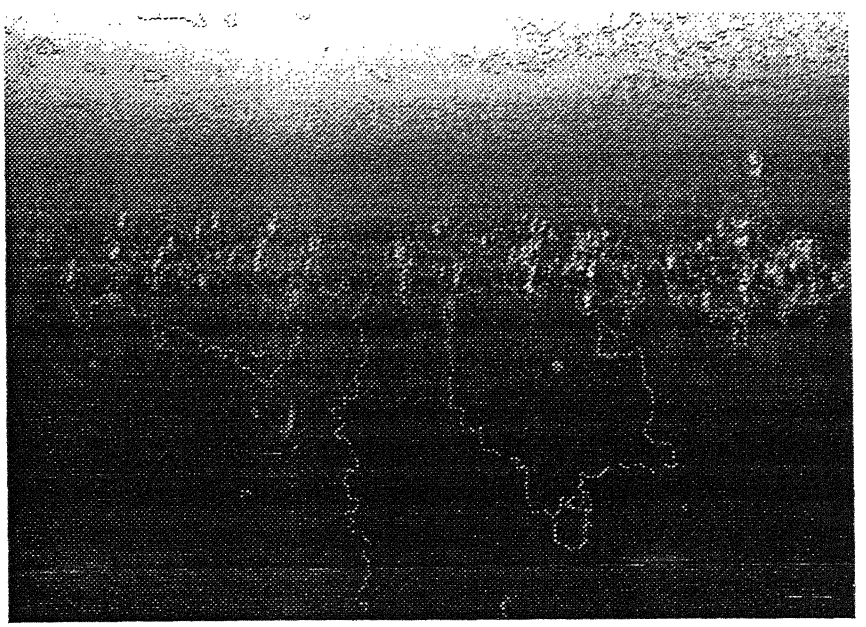


Figure 3.18: Watershed Segmentation with $h = 6$ and $n = 2$

Chapter 4

Conclusions and Future Scope

4.1 Conclusions

Watershed segmentation when combined with various preprocessing techniques is indeed a powerful segmentation. Using this transformation we are able to segment the various kinds of steel defects like pinch marks, roll marks, rust, black patches etc. The result of the segmentation is compared with that of manual groundtruth. Various parameters are tuned for segmenting different defects present in the steel images. A quantitative parameter (Error Index) is devised upon which the segmentation is evaluated and thus various parameters were chosen by making a good segmentation. This is a first step towards making an unsupervised segmentation.

4.2 Future Work

1. More detailed study of effect of various parameters on segmentation and making a learning algorithm for optimizing the parameters of the whole segmentation process such that a reasonably good segmentation is produced for all kinds of defects at a given lighting condition.

2. Study of image acquisition system and lighting conditions suitable for this segmentation

3. Study of unsupervised extraction of markers for various defects suitable for on-line implementation

4. Taking cues from motion, which results a more guided segmentation
5. Fixing a pre-processing procedure to eliminate noise and an image simplification method before segmentation.
6. Color watershed segmentation
7. Ability to segment multiple kinds of defects in the same image
8. Incorporation of domain knowledge like frequency of occurring of a particular kind of defect for better guided segmentation.

Appendix A

Fast Watersheds through Immersion Simulation

```
#define MASK  // Initial value of a threshold level
#define WSHED // Value of the pixels belonging to the watersheds
#define INIT  // Inital value of  im_o

- INPUT :  im_i, decimal image;
- OUTPUT:  im_o, image of the labeled watersheds;

* INITIALIZATIONS
    -   Value INIT is assigned to each pixel of im_o: im_o(p) = INIT;
    -   current_label = 0
    -   current_dist: integer variable
    -   im_d: work image (of distances), initalized to 0;
* Sort the pixels of im_i, in the increasing order of their gray values
    Let h_min and h_max designate the lowest and highest
    values respectively.
* For h = h_min to h_max {
        /* geodesic SKIZ of level h-1 inside level h */
    For every pixel p such that im_i(p) == h {
     // These pixels are accessed directly through the sorted array
```

```
 im_o(p)  = MASK;
 if there exists p' ∈ N_G(p) such that
    im_o(p') > 0 or im_o(p') == WSHED {
         im_d(p) = 1;
         fifo_add(p);
 }
}
current_dist = 1; fifo_add(fictitious_pixel);
repeat indefinitely {
   p = fifo_first();
   if (p == fictitious_pixel){
         if fifo_empty() == true then BREAK;
         else { fifo_add(fictitious_pixel);
                current_dist = current_dist + 1;
                p = fifo_first();
         }
   }
   For every pixelp' in N_G(p) {
         if im_d(p') < current_dist and im_o(p') > 0 or
             im_o(p') = WSHED){
             /* i.e., p' belongs to an already labeled basin or
                 to the watersheds  */
             if im_o(p') > 0 {
                 if im_o(p') == MASK or im_o(p) == WSHED then
                     im_o(p) = im_o(p');
                 else if im_o(p) ≠ im_o(p') then
                     im_o ( p ) = WSHED;
             }
             else if im_o(p) == MASK then im_o(p) = WSHED
         }
   }
}  // end of repeat indefinitely
```

```
    /* checks if new minima have been discovered */
    For every pixel p such that im_i(p) == h{
      im_d(p) = 0;  /* the distance associated with p is reset to 0 */
      if im_d(p) == MASK {
        current_label = current_label +1;
        fifo_add(p);
        im_o(p) = current_label;
        while fifo_empty() = false {
           p' = fifo_first();
           For every pixel p'' in N_G(p') {
             if im_o(p'') == MASK {
                fifo_add(p'');
                im_o(p'') = current_label;
             }
           }
        }
      }
    }  // end of minima discovery loop
  }    // end of For h_min to h_max
* End of algorithm
```

Bibliography

[1] D.H Ballard and C.M Brown. *Computer Vision*. Prentice Hall, New Jersey, 1982.

[2] P.J. Besl, E.J. Delp, and R. Jain. Automatic visual solder joint inspection. *IEEE Journal of Robotics and Automation*, RA-1(1):42–56, 1985.

[3] Suresh Bindinganavle R, Fundakowski Richard A, and Levitt Tod S. A real–time automated visual inspection system for hot steel slabs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(6):563–572, November 1983.

[4] Barry Brusey. Surface inspection technology - a visual experience. In *Proc. AISE Intl. Automated Surface Inspection Technology Conference*, 1999.

[5] Barry Brusey. Surface inspection technology - a visual experience. In *Proc. AISE Intl. Automated Surface Inspection Technology Conference*, 1999.

[6] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–697, November 1986.

[7] Arun Dalmia and Jakob Horstmann. From pixels to patterns: Design of an advanced classification engine. In *Proc. AISE Intl. Automated Surface Inspection Technology Conference*, 1999.

[8] Hon-Son Don, King-Sun Fu, C.R.Liu, and Wei-Chung Lin. Metal surface inspection using image processing techniques. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC–14(1):138–146, January/February 1984.

[9] G.N.Saridis and D.M. Brandin. Automatic surface inspection system for flat rolled steel. *Automatica*, 15:505–520, 1979.

[10] Robert M. Haralick and Linda G. Shapiro. Survey: Image segmentation techniques. *Computer Vision, Graphics and Image Processing*, 29:100–132, 1985.

[11] Robert M. Haralick, Stanley R. Sternberg, and Xinhua Zhuang. Image analysis using mathematical morphology. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(4), July 1987.

[12] Jinyang Jung and Kwang Jae Cho. Development of surface inspection system for cold rolled strip. In *Proc. AISE Intl. Automated Surface Inspection Technology Conference*, 1999.

[13] Tapas Kanungo. Documentation degradation models and a methodology for degradation model validation. Master's thesis, Center for Automation Research, University of Washington, Seattle, WA, USA, 1996. http://www.cfar.umd.edu/ kanungo/pubs/phdthesis.ps.Z.

[14] Tapas Kanungo and Robert M. Haralick. Character recognition using mathematical morphology. http://www.cfar.umd.edu/ kanungo/pubs/usps90-ocr.ps.Z.

[15] Tapas Kanungo and Robert M. Haralick. An automatic closed-loop methodology for generating character groundtruth for scanned documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, February 1999. http://www.cfar.umd.edu/ kanungo/pubs/autogt-pami.ps.Z.

[16] Ashok Kumar. Real–time based dsp identification system using content–based imaging techniques. Master's thesis, Dept of Electrical Engg, 1999.

[17] H. Lorin. *Sorting and Sort Systems (The System Programming Series)*. Addison-Wesley, Reading, MA, 1975.

[18] N. Maiti, U.B. Desai, A.K. Ray, and L.M. Gantayat. Mathematical morphology: An image processing tool in measurement of droplet size distribution in dropwise condensation. In *Indian Conference on Computer Vision, Graphics and Image Processing, 21-23, December, New Delhi, India*, 1998.

[19] Amitabha Mukerjee, Amitanand Nema, Sushmit Sen, and Sumana Gupta. An area-scan defect atlas of crm surface defects. In *Proc. AISE Intl. Automated Surface Inspection Technology Conference*, 1999.

[20] Gary L. Neiheisel, Robert J. Justice, Christopher L. Miller William W. Nagle, and Scott L. Miller. Hole detection at armco. In *Proc. AISE Intl. Automated Surface Inspection Technology Conference*, 1999.

[21] Amit Anand Nema. Defect atlas and suface defect identification in stell rolls. Master's thesis, Dept of Mechanical Engg, 1999.

[22] David Park. Detection the heart of classification. In *Proc. AISE Intl. Automated Surface Inspection Technology Conference*, 1999.

[23] Timo Piironen, Esko Strommer, Olli Silven, Toni Laitinen, and Matti Pietikaiinen. An automated visual inspection system for rolled metal surfaces.

[24] Reinhard Rinn, Scott A. Thomson, and John Torre Ralph Foehr, Friedrich Luecking. Parsytec hts-2, defect detection and classification through software vs. dedicated hardware. In *Proc. AISE Intl. Automated Surface Inspection Technology Conference*, 1999.

[25] Philippe Salembier and Montse Pardas. Hierarchical morphological segmentation for image sequence coding. *IEEE Transactions on Image Processing*, 3(5):639–651, September 1994.

[26] Sonka, Milan, H Lovac, Vaclav, and Boyle Richard. *Image processing, Analysis and Machine vision.* ITP publishers, Pacific Grove, 1999.

[27] Luc Vincent. Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms. *IEEE Transactions on Image Processing*, 2(2):176–201, April 1993.

[28] Luc Vincent and Perrie Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598, June 1991.

[29] Tychowsky V.W. From anomalies to defects: The challenge of classification in image processing application. In *Proc. AISE Intl. Automated Surface Inspection Technology Conference*, Florida, February 22-24 1999.

[30] Demin Wang. A multiscale gradient algorithm for image segmentation using watersheds. *Pattern Recognition*, 30(12):2043–2052, 1997.

A130911

Date Slip

This book is to be returned on the date last stamped.
